Domeinmodelleren

Programma Pandemisch Parate Informatievoorziening (PP-IV)

Versiebeheer

VERSIE	DATUM	STATUS	AUTEURS	OMSCHRIJVING
0.1	01-05-2024	Draft	Bas van der Raadt	Initiële opzet
0.2	16-05-2024	Draft	Atte Bootsma, Bas van der Raadt	Toevoeging introductie en context Hoofdstuk 1 t/m 5. Aanscherping en aanvulling voorbeeld domeinmodel
0.3	23-05-2024	Draft	Holke Visser	Review Hoofdstuk 1 t/m 5
0.4	31-05-2024	Finale draft	Holke Visser, Bas van der Raadt, Claudenir Morais Fonseca	Validatie van (meta)modellen op correct gebruik DEMO en OntoUML
0.8	06-06-2024	Finale draft	Atte Bootsma	Verwerken review-commentaar ICT-commissie LOI en PGIM
0.9	10-06-2024	Finale draft	Wilma Kruithof	Communicatie
1.0	11-06-2024	Definitief	Atte Bootsma, Bas van der Raadt	Review van Wilma Kruithof verwerkt in finale versie

Inhoudsopgave

1 Introductie

In de doelarchitectuur voor het pandemisch parate IV-landschap worden drie belangrijke uitgangspunten beschreven: schaalbaarheid, wendbaarheid en betrouwbaarheid.
Een belangrijk hulpmiddel bij het bereiken daarvan is de datacentrische aanpak die tot uiting komt in het domeinmodel voor de infectieziektebestrijding (IZB).

In dit document wordt beschreven wat een domeinmodel is, waarom het essentieel is om een schaalbaar, wendbaar en betrouwbaar IV-landschap te kunnen realiseren, hoe het IZB-domeinmodel is opgebouwd en hoe het tot stand komt. Tot slot geven we een uitgewerkt voorbeeld van één van de processen die in de infectieziektebestrijding plaatsvinden: Diagnostiek. Dit voorbeeld heeft als doel om de domeinmodelleringsmethodiek en de wijze van formeel specificeren van het IZB-domeinmodel te illustreren.

De GGD'en en het RIVM hebben niet eerder zo'n gezamenlijk model opgesteld voor één van de domeinen in de Publieke Gezondheid. De opzet van het model en van het totstandkomingsproces, zoals deze in dit document beschreven worden, zijn het resultaat van een traject dat in de afgelopen periode is doorlopen. Hierbij zijn zowel IZB- als IV-professionals betrokken geweest. Onderdeel van dat traject is dat we onze inzichten laten toetsen door externe deskundigen, bijvoorbeeld door de vakgroep Semantics, Cybersecurity and Services (SCS) van de Universiteit Twente, die onder andere gespecialiseerd is in semantisch modelleren.

Het model en de wijze waarop het tot stand komt zijn nog in ontwikkeling en zullen dat de komende periode ook blijven. Tijdens het programma zullen model en werkwijze verder worden uitgewerkt en, waar nodig, aangepast in samenwerking met alle betrokkenen (IZB-professionals, IV-professionals, ketenpartners, IV-leveranciers en anderen). Dat betekent ook dat de beschrijving in dit document onvermijdelijk een momentopname is.

2 Wat is een domeinmodel?

Een domeinmodel is een zo compleet mogelijke beschrijving van een domein. Voor het IZB-domeinmodel gaat het dan om het domein van de infectieziektebestrijding in Nederland, voor zover die wordt uitgevoerd door het RIVM en de GGD'en.

Het model kent twee doelstellingen of weergaves: de beschrijvende en de formele. Het model moet enerzijds begrijpelijk en bruikbaar zijn voor uitvoering en organisatie van activiteiten in het IZB-domein. Paragraaf 6.4 bevat de beschrijvende weergave van het model. Daarnaast moet het model formeel en exact zijn voor ontwerp en realisatie van de informatievoorziening. Dit is de formele weergave van het model. De formele weergave wordt uitgebreid toegelicht in paragraaf 6.7.

In het domeinmodel wordt het IZB-werkveld vanuit verschillende perspectieven belicht, onder andere vanuit het perspectief van de activiteiten, de informatie, de techniek of de wet- en regelgeving. Dit is vergelijkbaar met de menselijke anatomie, waarbij het menselijk lichaam vanuit verschillende perspectieven wordt bestudeerd (denk aan: hart- en vaatstelsel, spijsverteringsstelsel, etc.). Elk perspectief kent zijn eigen beschrijving, met een eigen manier van vastleggen. Zo ontstaan verschillende deelmodellen, die samen zorgen voor één consistente en samenhangende weergave van het domein. Daarom hanteren we de term 'domeinmodel' in enkelvoud, terwijl het eigenlijk een verzameling van modellen is.

De volledige definitie van het IZB-domeinmodel is:

- Een gestructureerde kennisbank over en voor de gezamenlijke infectieziektebestrijding in de:
    - reguliere
    - en pandemische fase
- Met als doel:
    - de verbetering van de communicatie en samenwerking in de IZB
    - de realisatie van een pandemisch parate, wendbare, datacentrische, modulaire informatievoorziening, die dit ondersteunt
- Waarin:
    - informatie, betekenis, bedrijfsregels en datatoegang centraal staan en eenduidig zijn vastgelegd
    - in samenhang met processen enerzijds
    - en applicatiefuncties anderzijds
- Bestaande uit:
    - verschillende samenhangende weergaves voor verschillende stakeholders
- Ontwikkeld:
    - in verschillende modelleertalen
    - gebruikmakend van verschillende soorten modelleer disciplines
    - door verschillende ontwikkelaars met verschillende methoden en verschillende vaktermen

3 Waarom domeinmodelleren?

Een aantal voorbeelden die het belang van het domeinmodel illustreren:

Tijdens de pandemie bleek dat bijspringen in een andere regio mij als ervaren IZB verpleegkundige zeker 2 dagen **inwerktijd** kostte. Door eenduidige definitie van werkprocessen en informatie, kan ik naadloos bijspringen in andere GGD-regio's bij uitbraken.

"Het IZB domeinmodel geeft de IZB-keten **eenheid van taal**; bij een dreigende pandemie hoeven we geen discussie te voeren over de betekenis van de informatie die we van elkaar nodig hebben." Voorbeeld: Eenheid van taal ABR

Surveillancegegevens leveren onduidelijkheid in surveillance door **verschillende definities** van possible, probable, confirmed. Ondubbelzinnige afspraken voorkomen dit.

De V&V keten heeft ervaring opgedaan met **modelgebaseerde informatie-uitwisseling**. Hierdoor zijn administratieve lasten verminderd en databeschikbaarheid verbeterd (KIK-V).

Het ZonMw Harmony project combineert COVID-vaccinatie datasets. Ondanks goede afspraken vooraf over het protocol kostte het anderhalf jaar om tot een bruikbare **geharmoniseerde dataset** te komen. Een IZB-domeinmodel verbetert dit door o.a. informatiedefinities en data-governance vast te leggen. Onderzoeksresultaten zijn dan sneller beschikbaar en onderzoeksvragen worden sneller beantwoord waardoor meer mensen beter beschermd kunnen worden bij een COVID-19 infectie.

"Labuitslagen komen vrijwel altijd via e-mail of PDF binnen en moeten worden **overgetypt**. Dankzij eenheid van taal in de labs kon tijdens de pandemie heel snel een standaard orderbericht voor COVID-testen worden gemaakt, waardoor 60 labs konden worden aangesloten"

De toepasselijke wetgeving & AmvB's zijn als grondslagen opgenomen in het IZB-domeinmodel en dragen bij aan de **correcte toegangsverlening** tot IZB-informatie

Figuur 1: Voorbeelden die het belang van het domeinmodel illustreren

Veel van bovenstaande issues vinden hun oorsprong in het volgende:

- Het IV-landschap van de infectieziektebestrijding bestaat, net als veel andere landschappen in de zorg en de publieke gezondheid, uit een groot aantal verschillende applicaties die elk op hun eigen manier data opslaan waardoor deze lastig herbruikbaar is buiten de applicatie.
- Daarnaast heeft elke GGD een grote vrijheid om die applicaties in te richten, waardoor gegevens uit dezelfde velden een verschillende betekenis kunnen hebben.
- Gebruikte termen zijn vaak niet eenduidig vastgelegd (vraag 20 IZB-professionals naar een definitie van het begrip Surveillance en je krijgt 20 verschillende antwoorden).
- Er zijn slechts beperkt (meestal alleen op regionaal niveau) afspraken gemaakt over geüniformeerde uitvoeringsprocessen waardoor er een grote variatie bestaat in de uitvoering van de landelijke richtlijnen.
- Het IZB-werkveld valt niet binnen één organisatie, maar omvat een ketensamenwerking tussen meerdere organisaties (o.a. GGD'en, GGD GHOR Nederland, RIVM, etcetera). Vaak hanteren organisaties eigen terminologie en kijken anders aan tegen de informatie die zij verwerken en uitwisselen.
- Een beperkt gebruik van standaarden voor opslag, uitwisseling en definitie van informatie.

Het realiseren van een pandemisch paraat, schaalbaar, wendbaar en betrouwbaar landschap is alleen maar mogelijk als die problemen worden opgelost en er een gedeeld model is waarin onder andere processen, data en betekenis zijn vastgelegd. Dit domeinmodel, dat door ketenpartners gezamenlijk gemaakt en beheerd wordt, helpt in het creëren van een eenheid van taal en betekenis en eenduidige informatie-uitwisseling. En dat leidt weer tot minder spraakverwarring en miscommunicatie tussen ketenpartners. Met het model kan een datacentrisch landschap worden gerealiseerd, waarin alle systemen gebruik gaan maken van hetzelfde informatiemodel. Als de betekenis exact en eenduidig is bepaald, kunnen de systemen deze correct op eenduidige wijze verwerken. Hierdoor krijgt men toegang tot kwalitatief hoogwaardige informatie. Ook wordt het mogelijk om zo snel wijzigingen in verwerking en definitie van de informatie door te voeren.

Het domeinmodel maakt het, kortom, mogelijk om te kunnen redeneren over ons eigen werkveld:

- Door het eenduidig vastleggen van een gedeeld beeld van de werkelijkheid.
- Zodat er meer duidelijkheid is over betekenis, herkomst en totstandkoming van data.
- Om de kwaliteit van de eigen informatiehuishouding beter op orde te hebben.
- Om ieder te laten beschikken over dezelfde hoogwaardige gegevens.
- Om het delen van gegevens met anderen beter (eenduidiger, eenvoudiger) te laten verlopen.
- Zodat we beter met elkaar kunnen communiceren.
- Zodat we beter kunnen samenwerken.
- Zodat de werking van de organisatie meer inzichtelijk is.
- Zodat we beter onze gemeenschappelijk doelen/resultaten kunnen behalen.
- Zodat we eenvoudiger over kunnen gaan van een regionale naar een landelijke aansturing en weer terug.
- Zodat we de toegang tot data kunnen verbeteren.

3.1.1 Waarom formeel domeinmodelleren?

De reden dat we zoveel aandacht en energie investeren in het modelleren van de informatie is de datacentrische aanpak die in de doelarchitectuur (zie Bijlage 02 - Doelarchitectuur IV voor infectieziektebestrijding editie 2024Q1) is gekozen. We kunnen niet meer volstaan met het specificeren van algemene functionele eisen. We zullen vanwege de gekozen datacentrische aanpak in meer detail moeten beschrijven hoe de informatie verwerkt moet worden.

Waarom is er dan voor gekozen om een gedetailleerd informatiemodel formeel in OntoUML te modelleren. Waarom kunnen we niet volstaan met een simpel ERD of UML class-model?

3.1.1.1 Complexiteit

De eerste reden is dat het domein van de infectieziektebestrijding complex is. Het informatiedomein van de IZB bestaat, naast het jargon dat te maken heeft met de bestrijding zelf, ook uit medische, biologische en sociale begrippen. De diversiteit en ordening van informatie in deze domeinen is veel uitgebreider dan in andere domeinen, zoals bijvoorbeeld het bezorgen van pakjes. De termen zijn ook nog eens aan verandering onderhevig. In de reguliere fase is er verandering als gevolg van medisch wetenschappelijk onderzoek en sociale ontwikkelingen. Tijdens een uitbraak of pandemie kunnen veranderingen in een stroomversnelling komen of tot radicaal nieuwe informatiebehoeften leiden. De complexe aard (diversiteit, ordening, dynamiek) van het werkgebied maakt dat we zeer scherp op de semantiek moeten zijn om ondubbelzinnig met elkaar te kunnen communiceren.

3.1.1.2 Internationale samenwerking

De tweede reden om de betekenis formeel en exact te modelleren met OntoUML is het feit dat infectieziektebestrijding een internationale aangelegenheid is. Ziektes kunnen zich snel over de wereld verspreiden, omdat we met elkaar verbonden zijn door allerlei vormen van transport. Het ondubbelzinnig vastleggen van de betekenis is een noodzakelijke voorwaarde om informatie uit te kunnen wisselen met mensen uit andere landen met andere talen.

3.1.1.3 Toepassing informatiestandaarden

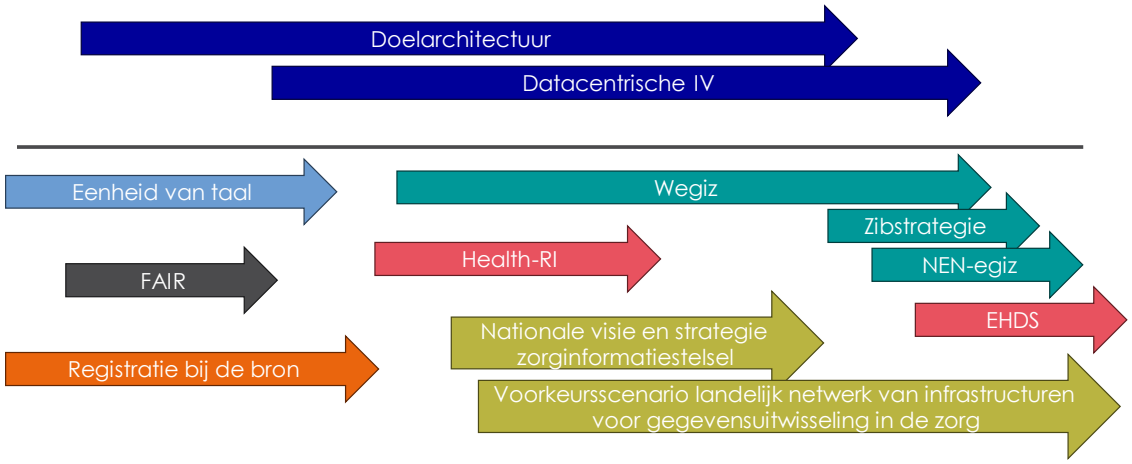
Het toepassen van informatiestandaarden helpt bij het uitwisselen van gegevens, zowel nationaal als internationaal. Echter er zijn vele informatiestandaarden die elkaar overlappen. Niet alle partijen passen de standaarden toe. Standaarden kennen meerdere versies en worden niet altijd consequent toegepast. Dat is de derde reden dat het nodig is om de eigen informatiebehoefte systematisch, formeel en exact vast te leggen. Om zo de comptabiliteit van de eigen informatiebehoefte met een in de praktijk toegepaste standaard te kunnen bepalen.

3.1.1.4 Data integratie

Naast de communicatie en het uitwisselen van informatie, zijn er nog andere redenen waarom een exacte en formele definitie van de informatie een noodzakelijk voorwaarde is. Een vierde reden is dat een formeel betekenismodel van de informatie helpt bij het integreren van verschillende databronnen, zoals data uit verschillende systemen, legacy systemen en externe databronnen. Het is bruikbaar voor traditionele vormen van dataverwerking zoals: datawarehousing, en business intelligence. Maar het is ook nuttig voor moderne vormen van dataverwerking zoals: knowledge graphs en Large Language Models.

3.2 Ontwikkelingen in de keten

De IZB is geen eiland. Bij de totstandkoming van de doelarchitectuur is niet alleen gekeken naar de lessen uit de COVID-pandemie, maar ook rekening gehouden met huidige en toekomstige ontwikkelingen in beleid, met name de nationale visie en strategie van op het zorginformatiestelsel, NVS, en wetgeving, zoals de Wegiz, EHDS en Wpg. Een datacentrisch landschap, dat gebaseerd is op een domeinmodel (waarin betekenissen helder zijn vastgelegd) is volledig in lijn met die ontwikkelingen. Het is ook in lijn met andere ontwikkelingen binnen de Zorg-ICT, zoals: Eenheid van taal, FAIR, Registratie bij de bron en de Zib-strategie.

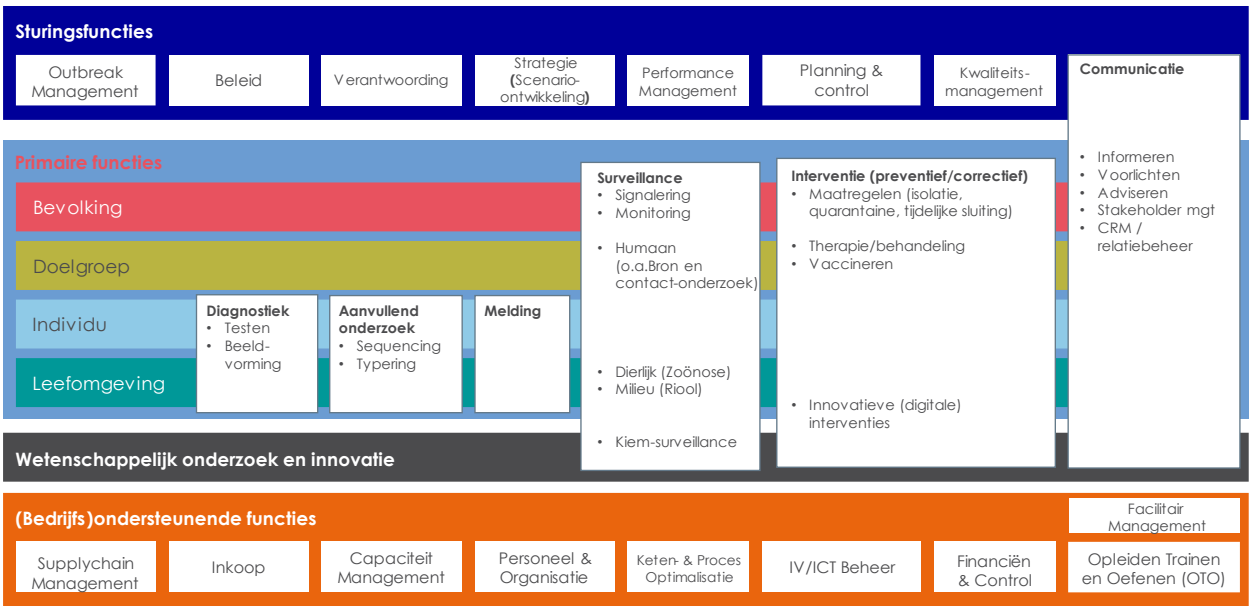


Figuur 2: Relevante ontwikkelingen in de keten

4 Uit welke elementen bestaat het domeinmodel?

4.1 Bedrijfsfunctiemodel

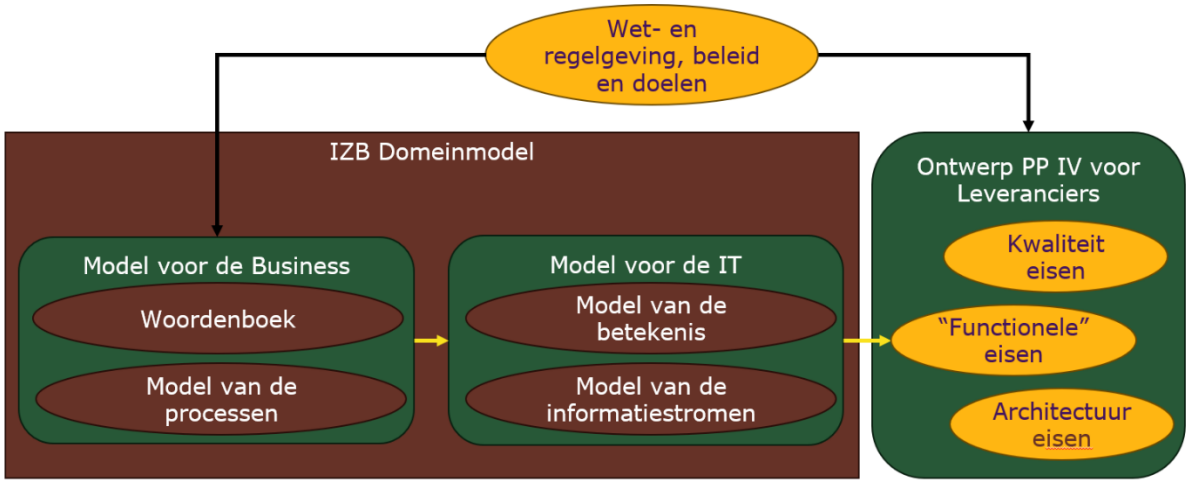
De belangrijkste bedrijfsfuncties zijn vastgelegd in de doelarchitectuur. Deze functies worden in het domeinmodel uitgewerkt:



Figuur 3: Bedrijfsfunctiemodel doelarchitectuur

4.2 Indeling

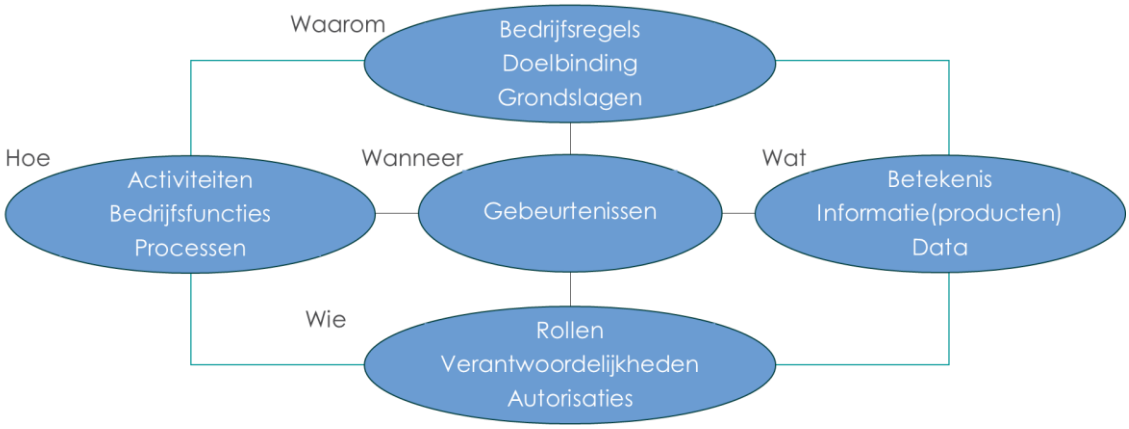
Het domeinmodel bestaat, zoals hierboven al vermeld, uit een beschrijvende weergave en een formele weergave. De belangrijkste aspecten daarvan worden weergegeven in Figuur 4. De inhoud van de beschrijvende weergave wordt bepaald door IZB-professionals. In de formele weergave wordt die inhoud omgezet in formele architectuurmodellen. Een uitwerking daarvan is te vinden in paragraaf 6.7. Het domeinmodel als geheel vormt de basis voor het ontwerp van de IV-voorzieningen.



Figuur 4: Belangrijkste aspecten van het domeinmodel

Een zo volledig mogelijke beschrijving van het IZB-domein bevat, uiteraard, meer dan alleen betekenissen en processen. Ambitie is om in het model de volgende aspecten uit te werken (zie Figuur 5):

1. Waarom
2. Hoe
3. Wanneer
4. Wat
5. Wie



Figuur 5: Uitwerking van het domeinmodel

4.2.1 Waarom?

4.2.1.1 Bedrijfsregels

Een bedrijfsregel is een verklaring die een bepaald aspect van het bedrijf definieert of beperkt. Het is bedoeld om de bedrijfsstructuur te bevestigen of om het gedrag van het bedrijf te controleren of te beïnvloeden (bron: Rules, R. Ross). Bijvoorbeeld: Voor het zetten van een vaccinatie bij een kind onder de 12 jaar is toestemming nodig van de ouder(s).

4.2.1.2 Doelbinding

De doelbinding legt vast, met welk doel activiteiten worden uitgevoerd en informatie wordt vastgelegd. Dit zorgt ervoor dat alleen gegevens worden verwerkt en verzameld voor een bepaald, omschreven en gerechtvaardigd doel en dat achteraf altijd achterhaald kan worden met welk doel informatie tot stand is gekomen.

4.2.1.3 Grondslagen

De grondslagen bevatten alle wet- en regelgeving die van toepassing kan zijn op de activiteiten en de informatie in het domein. Hier wordt bijvoorbeeld vastgelegd of er informatie tot stand is gekomen in het kader van een Wpg-activiteit of dat de WGBO van toepassing is.

4.2.2 Hoe

4.2.2.1 Activiteiten

Een activiteit is een afgebakende stap in een proces. Bijvoorbeeld: Het afnemen van het monster in het diagnose-proces.

4.2.2.2 Bedrijfsfuncties

Een bedrijfsfunctie beschrijft datgene wat gedaan moet worden om een bedrijfsdoelstelling te behalen, onafhankelijk van de inrichting van de organisatie en haar processen. Het representeert (een clustering van) verantwoordelijkheden en taken die soortgelijke kennis, vaardigheden, competenties en/of middelen vragen. Voorbeelden zijn Diagnostiek, Interventie, Kwaliteitsmanagement of Inkoop.

4.2.2.3 Processen

Een bedrijfsproces is een geordende reeks procestappen die door één organisatie wordt uitgevoerd, waarna het resultaat tot stand is gekomen. Het transformeert voor het bedrijf relevante inputs in relevante outputs. Een voorbeeld is het diagnose-proces.

4.2.3 Wanneer

4.2.3.1 Gebeurtenissen

Een gebeurtenis is het moment waarop de toestand van een (bedrijfs/informatie)object verandert. Een voorbeeld is de overgang van ‘Dossier geopend’ naar ‘Dossier gesloten’.

4.2.4 Wat

4.2.4.1 Betekenis

De betekenis van gebruikte termen wordt vastgelegd in een woordenboek, zodat voor alle betrokkenen helder is wat met welke term bedoeld wordt. Daarbij wordt rekening gehouden met een verschuiving van betekenis in de tijd: Een term kan in de loop van de tijd meerdere betekenissen hebben. Ook leggen we de betekenis vast in formele talen zodat er geautomatiseerd gecombineerd kunnen worden en er geautomatiseerd analyses op gedaan kunnen worden.

4.2.4.2 Informatie(producten)

Een digitaal of fysiek product dat informatie aanbiedt aan een gebruiker wordt een informatieproduct (IP) genoemd. Een informatieproduct bestaat uit een of meerdere informatie-elementen. Een voorbeeld is een medisch dossier of een epidemiologisch rapport.

4.2.4.3 Data

Data of een dataobject is een op zichzelf staand geheel van gegevens met een eigen identiteit. Een voorbeeld is een testuitslag of een e-mailbericht.

4.2.5 Wie

4.2.5.1 Rollen

Een rol is een combinatie van samenhangende bevoegdheden en verantwoordelijkheden die een medewerker meestal tijdelijk op zich neemt en die afwisselend door verschillende personen kunnen worden vervuld. Een voorbeeld is een arts die bevoegd is om medische handelingen uit te voeren of een administratief medewerker die een afspraak voor een spreekuur inplant.

4.2.5.2 *Verantwoordelijkheden*

In de verantwoordelijkheden wordt vastgelegd welke activiteiten door welke rollen mogen worden uitgevoerd. Een voorbeeld is een arts die een diagnose mag vastleggen.

4.2.5.3 *Autorisaties*

Een autorisatie is een expliciete toestemming afgegeven door een autoriteit aan een rol om een handeling uit te voeren. Uitgangspunt is: nee – tenzij. Dat wil zeggen dat handelingen alleen mogen worden uitgevoerd als daar een expliciete autorisatie voor bestaat.

4.3 Domeinmodel principes

Bij het ontwerpen en vullen van het domeinmodel wordt een aantal principes gehanteerd die onder andere voortkomen uit de doelarchitectuur en het daarvan afgeleide conceptuele model:

Losjes gekoppeld: de functionaliteit is niet door code gekoppeld aan het dataopslagsysteem. Beide lagen kunnen worden vervangen zonder dat dat aanpassingen aan de andere laag vraagt (en dus ook zonder dat informatie verloren gaat).

Eén enkel: niet één datamodel per systeem, maar één eenduidig model voor het hele domein, in de toekomst mogelijk uitgebreid naar andere domeinen.

Eenvoudig: niet vele concepten per systeem die niet gebruikt worden of niet (of met een andere betekenis) voorkomen in meerdere systemen maar centrale concepten waarmee de hele organisatie beschreven kan worden.

Uitbreidbaar: niet een nieuwe applicatie maken voor een nieuwe taak maar uitbreiden van bestaande functionaliteit.

Federeerbaar: niet data verplaatsen maar een vraag over meerdere systemen tegelijk heen kunnen stellen.

Universeel: niet verschillende datastructuren transformeren maar alle datastructuren opdelen in de elementaire semantische bouwblokken.

Gedeeld: niet data kopiëren maar data delen.

Direct implementeerbaar : niet een apart conceptueel, logisch en technisch model maar één semantisch datamodel.

Veiligheid in het ontwerp: geen 'toegang tenzij het verboden is' maar alleen toegang als dat expliciet is toegestaan (conform het zogenaamde 'zero-trust' principe).

Privacy in het ontwerp: Gegevensbescherming als onderdeel van het ontwerp en niet als een functionele laag daar bovenop.

4.4 Bronnen, referentiemodellen en standaarden

Bij het opstellen van het domeinmodel wordt zoveel als mogelijk gebruik gemaakt van referentiemodellen en bestaande standaarden. De belangrijkste daarvan zijn:

- De doelarchitectuur voor de IZB
- De IZB-domeinarchitectuur
- Vijf-lagen model (uit Nederlandse Overheid Referentie Architectuur)

- Specificatiecanvas van het gezondheidsinformatiestelsel en de nationale terminologieserver (Nictiz)
- Regels en richtlijnen van het European Centre for Disease Prevention and Control (ECDC)
- Public Health Conceptual Datamodel van de Centres for Disease Control and Prevention van de USA
- Principes uit de gebruikte architectuurtalen en methodes (ArchiMate, UML, OntoUML, DEMO, BPMN)
- Zorginformatiestandaarden zoals OpenEHR, SNOMED en LOINC

5 Hoe komt het IZB-domeinmodel tot stand?

Zoals in een vorig hoofdstuk beschreven, is het IZB- domeinmodel in feite een kennismodel van de infectieziektebestrijding in Nederland. Met name dat deel dat door de GGD'en en het RIVM wordt uitgevoerd. IZB-professionals zijn de kennishouders (materiedeskundigen) van dit werkveld. Het IZB-domeinmodel kan alleen tot stand komen met de hulp van IZB-professionals. Daarom is een gebruikersorganisatie opgezet waarin IZB-professionals samenwerken met IV-professionals (analisten, architecten en informatiemanagers).

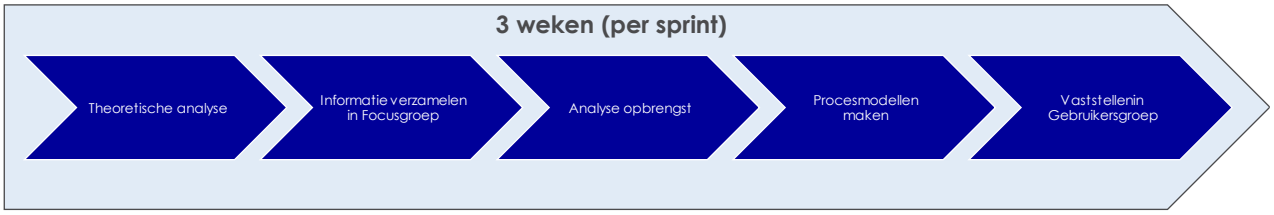
5.1 Focusgroepen en gebruikersgroepen

De gebruikersorganisatie bestaat in de kern uit focusgroepen en een gebruikersgroep:

- Focusgroepen: Groepen die samen een ontwerp van een deelproces maken, bijvoorbeeld het procesmodel van de bedrijfsfunctie 'Melden'.
  Een focusgroep bestaat uit:
  - IZB-professionals (de samenstelling kan per onderwerp verschillen) uit de verschillende regio's in het land.
  - Eén of meer analisten.
  - Eventueel architecten.
- De gebruikersgroep: In deze groep komen de modellen vanuit de focusgroep samen en worden ze gevalideerd.
  De gebruikersgroep bestaat, naast de voorzitter, uit IZB-professionals, zo mogelijk evenredig verdeeld over de vijf IZB-regio's en evenredig verdeeld over de disciplines in de IZB:
  - Artsen
  - Verpleegkundigen
  - Deskundigen infectiepreventie (DIP)
  - Epidemiologen
  - Waar nodig aangevuld met ondersteunende functies als de doktersassistenten en administratiekrachten.

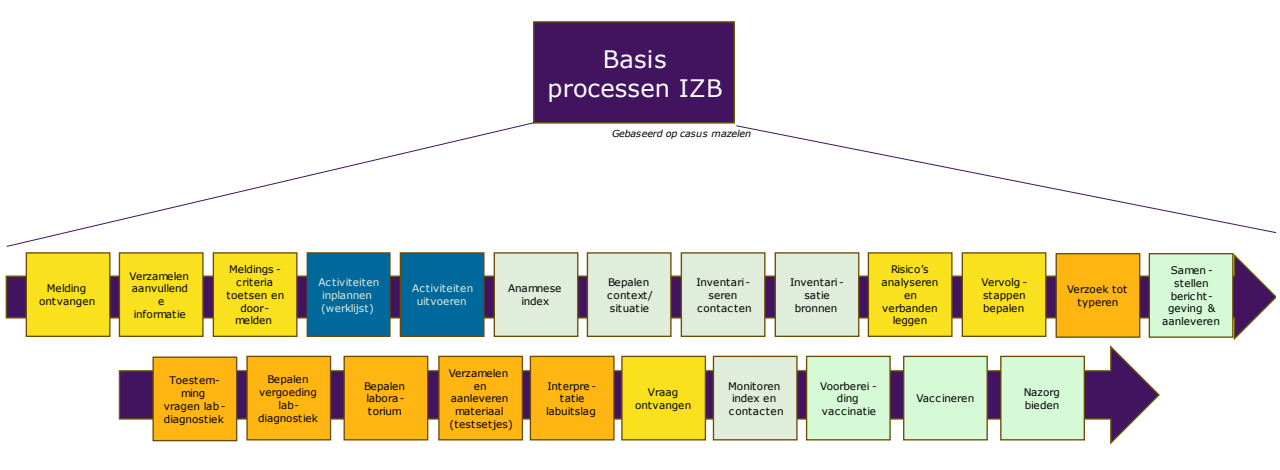
5.2 Procesritme

Het proces verloopt, conform het ritme van het programma, in sprints van drie weken, waarbij steeds een deel wordt uitgewerkt en daarna wordt vastgesteld:



Figuur 6: Procesritme van het domeinmodelleren

5.3 Deelprocessen

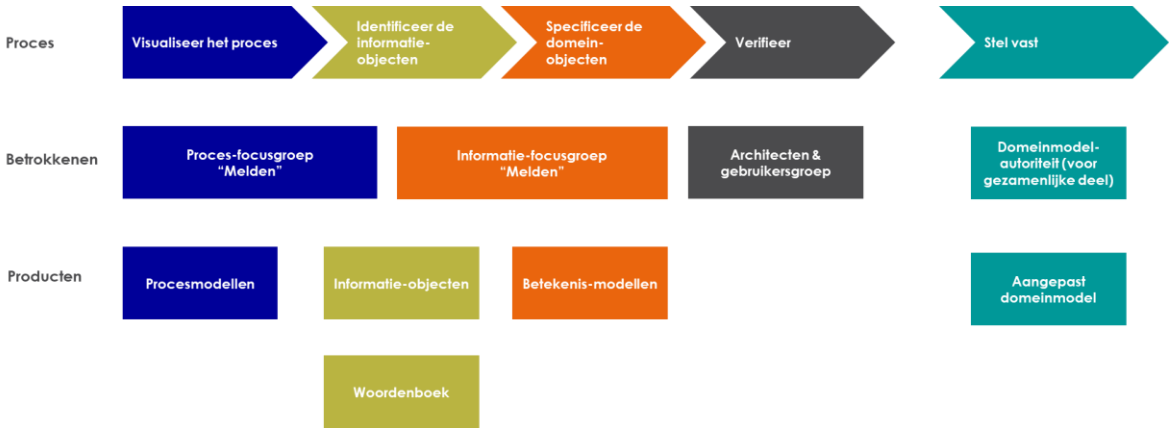


Figuur 7: Basisprocessen IZB

Naar aanleiding van een eerste analyse van het bedrijfsfunctiemodel zijn voor de reguliere IZB 23 deelprocessen geïdentificeerd (zie Figuur 7) die worden uitgewerkt op zo'n wijze dat de uitwerking ook in een pandemische situatie bruikbaar is. In de komende periode worden de deelprocessen bepaald die vervolgens worden uitgewerkt voor de modellering van een aantal specifieke IZB-functies als grootschalig klantcontact en grootschalig vaccineren.

5.4 Geleidelijke vulling

Het domeinmodel bestaat uit een aantal deelmodellen (bijvoorbeeld het procesmodel en het informatiemodel). Bij modelleren van de deelprocessen van de IZB is ervoor gekozen om niet alle deelmodellen in één keer uit te werken, maar te starten met het vullen van het procesmodel. Dit model is immers voor IZB-professionals het meest herkenbaar vanuit hun dagelijkse werk. Vanuit dit uitgewerkte procesmodel zullen vervolgens per deelproces de overige modellen worden gevuld (zie Figuur 8).



Figuur 8: Uitwerking modellen per deelproces

5.5 Van generiek naar specifiek

Dezelfde gefaseerde aanpak geldt voor het uitwerken van de verschillende deelprocessen zelf. Dat wil zeggen dat de modellering per deelproces zich in eerste instantie richt op het opstellen van een IZB-basismodel, waarin dat deel van processen wordt vastgelegd dat (vrijwel) altijd worden toegepast.

In een tweede en eventueel derde ronde worden daar vervolgens de uitzonderingen aan toegevoegd die alleen optreden bij specifieke situaties (bijvoorbeeld: Een GGD-regio met een grote haven of een internationaal vliegveld) of specifieke ziektes.

5.6 Bouwbare specificaties

Uiteindelijk moet het uitgewerkte model in een proces met de partner die de functionaliteit gaat realiseren worden doorontwikkeld naar bouwbare specificaties. Het model van focus- en gebruikersgroepen zal ook hiervoor worden ingezet. De groepen zullen dan worden aangevuld met één of meer vertegenwoordigers van deze marktpartij(en).

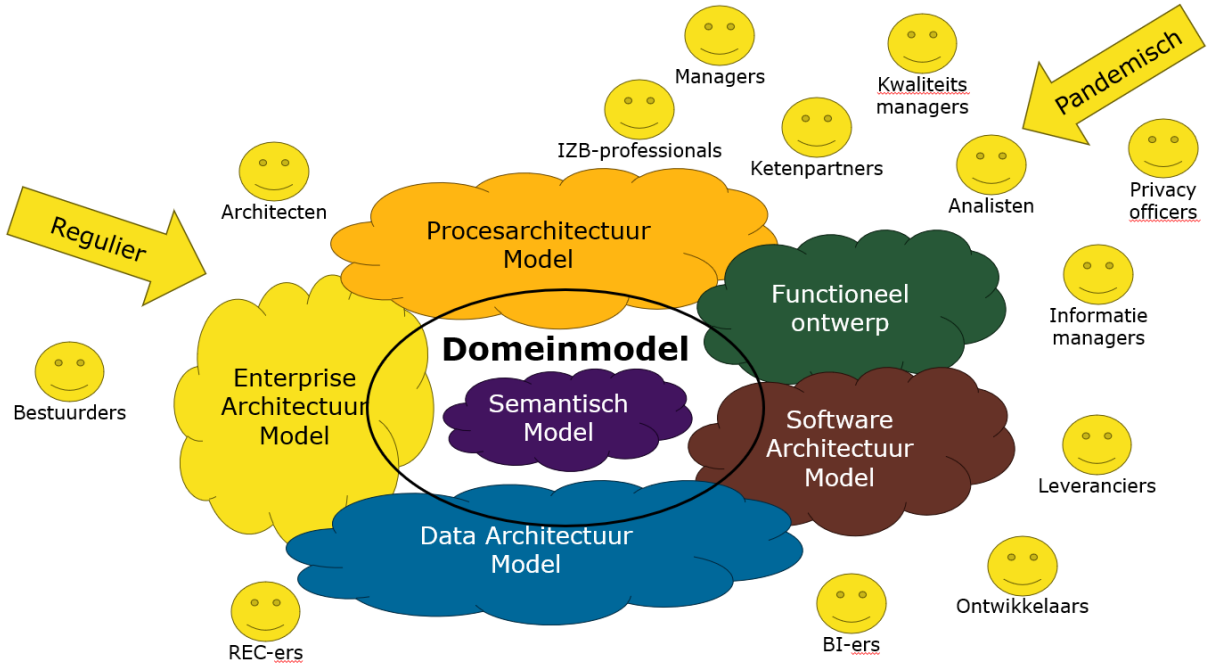
6 Formele uitwerking van het domeinmodel

6.1 Inleiding

In dit hoofdstuk beschrijven we onze huidige inzichten over de wijze waarop het domeinmodel zal worden vastgelegd. Dat doen we aan de hand van een uitgewerkt voorbeeld van één van de sub-domeinen van de IZB: Diagnostiek.

De nadruk ligt daarbij op het gebruik van de combinatie van modelleertalen en conventies in het model.

Deze uitwerking van een mini-domeinmodel van Diagnostiek is in eerste instantie geschreven voor potentiële partners die deelnemen aan de aanbesteding voor een nieuw functioneel platform voor de IZB, zodat zij kunnen reflecteren op de leesbaarheid van het model.



Figuur 9: Domeinmodel als set van samenhangende modellen die met elkaar het IZB-domein beschrijven

6.2 Leeswijzer

In de uitwerking van het domeinmodel wordt gebruik gemaakt van een combinatie van meerdere modelleertalen en methodieken, namelijk ArchiMate, DEMO en OntoUML. In paragraaf 6.3 wordt eerst kort toegelicht waarom deze combinatie wordt toegepast. In het vervolg van 6.3 worden de drie talen/

methodieken individueel toegelicht. In het domeinmodelleren wordt een onderscheid gemaakt tussen 'beschrijvend' modelleren en 'formeel' modelleren. In paragraaf 6.4 wordt eerst een 'beschrijvend' model van de 'activiteiten' die vallen binnen het Diagnostiek onderdeel van het IZB-domein gegeven in ArchiMate. Vervolgens wordt in paragraaf 6.5 beschreven hoe invulling wordt gegeven aan het 'formeel' modelleren door middel van een combinatie van DEMO en OntoUML. In paragraaf 6.6. wordt de mapping van 'beschrijvend' (ArchiMate) naar 'formeel' (combi van DEMO en OntoUML) toegelicht. Paragraaf 6.7 bevat alle formele specificaties van zowel de 'activiteiten' als de 'informatie' die vallen binnen het voorbeeld van Diagnostiek als onderdeel van het IZB-domein.

6.3 Modelleertalen

De in dit document gebruikte modelleertalen zijn:

- ArchiMate® 3.2 (https://pubs.opengroup.org/architecture/archimate3-doc/)
- DEMO (https://nl.wikipedia.org/wiki/DEMO)
- OntoUML (https://en.wikipedia.org/wiki/OntoUML)

Deze talen dienen ieder een eigen doel en zijn complementair aan elkaar:

- De **ArchiMate**-taal wordt gebruikt om de 'activiteiten' die vallen binnen het IZB-domein informeel te beschrijven. De ArchiMate taal heeft als voordeel dat deze breed bekend is binnen de digitale architectuurgemeenschap. Daarbij zijn ArchiMate platen leesbaar voor een grotere groep belanghebbenden. Een nadeel van de ArchiMate taal is het informele karakter en de beperkte mogelijkheden om o.a. informatiemodellen voldoende duidelijk uit te werken.
- De **DEMO**-methodiek wordt gebruikt om de 'activiteiten' die vallen binnen het IZB-domein formeel te specificeren. Deze methodiek heeft als voordeel dat deze leidt tot een zeer duidelijke en gestructureerde weergave van de elementaire procesbouwblokken van een organisatie. Een nadeel van DEMO is dat de modellen minder geschikt zijn voor een breed publiek.
- De **OntoUML**-taal wordt gebruikt om de 'informatie' die een rol speelt binnen het IZB-domein op een formele wijze te specificeren. OntoUML is een ontologische verdieping op de Unified Modeling Language (UML) en biedt daar mee twee voordelen. Ten eerste zijn OntoUML modellen zeer duidelijk en gestructureerd opgebouwd, waardoor de betekenis en structuur van informatie op een eenduidige wijze is vastgelegd. Ten tweede is er een directe link met UML als de facto standaard software ontwerptaal, waardoor de vertaling van de ontologie / het informatiemodel naar het implementatieniveau makkelijk te maken is.

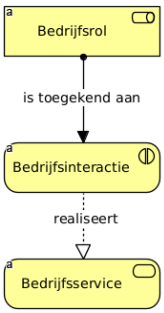
In het resterende deel van paragraaf 6.3 worden de drie talen/methodieken individueel toegelicht aan de hand van voorbeelden die niet iets te maken hebben met het Diagnostiek sub-domein.

6.3.1 ArchiMate

ArchiMate is een zeer rijke architectuurmodelleringstaal met vele soorten elementen die verdeeld zijn over meerdere architectuurlagen. Het is niet het doel van dit document om deze taal in zijn volledigheid te beschrijven. Mede omdat in de uitwerking van het voorbeeld domeinmodel in dit document gebruik gemaakt wordt van een subset van de elementen die de bedrijfslaag van de ArchiMate taal bevat (zie ook Figuur 10):

- **Bedrijfsrol:** vertegenwoordigt de verantwoordelijkheid voor het uitvoeren van specifiek gedrag, waaraan een actor kan worden toegewezen, of de rol die een actor speelt in een bepaalde actie of gebeurtenis. Voorbeelden hiervan zijn: `Receptionist` en `Klant`.
- **Bedrijfsinteractie:** vertegenwoordigt een eenheid van collectief gedrag dat wordt uitgevoerd door (een samenwerking van) twee of meer *bedrijfsrollen*. Voorbeelden hiervan zijn: `Inchecken klant` en `Beantwoorden klantvraag`.

- **Bedrijfsservice:** vertegenwoordigt expliciet gedefinieerd gedrag dat een *bedrijfsrol* blootstelt aan zijn omgeving. Voorbeelden hiervan zijn: `Beschikbaar stellen accommodatie` en `Verzorgen diner`.



Figuur 10: Extract uit het metamodel van ArchiMate

Wat hierin opvalt is dat in dit document geen gebruik wordt gemaakt van het 'bedrijfsobject' artefact van ArchiMate om het 'beschrijvende' perspectief op het 'Informatie' onderdeel van het domeinmodel te beschrijven. In het uiteindelijke domeinmodel zal dit wel worden beschreven, maar omwille van het beperken van de omvang van dit voorbeeld is ervoor gekozen dit niet in dit document op te nemen. In dit document wordt alleen de 'formele' weergave van het 'informatie' onderdeel weergegeven in de informatiemodelleringstaal OntoUML.

Het voordeel van ArchiMate is dat de notatiewijze breed bekend is binnen de digitale architectuurgemeenschap. Een nadeel is echter het minder formele karakter van de taal. Dit is het gevolg van de wat onduidelijke definities van de elementen van de ArchiMate. Daardoor is het onvoldoende geschikt om een formele structuur aan te brengen in het 'Activiteiten' onderdeel van het domeinmodel.

6.3.2 DEMO

Daarom wordt in het formeel specificeren van de 'activiteiten' gebruik gemaakt van de principes van de DEMO-methodiek. De DEMO-methodiek kent ook zijn eigen notatiewijze. In dit document is er echter voor gekozen om de DEMO-notatiewijze niet te gebruiken in de modellen. We gebruiken de ArchiMate- en OntoUML-notatiewijzen om de DEMO-concepten te visualiseren voor respectievelijk de beschrijvende specificaties. De lezer wordt zo niet belast met drie verschillende notatiewijzen. In deze paragraaf worden de basisprincipes en concepten van DEMO toegelicht. Het volgende document is hierbij als basis gebruikt:

Begrijpen en Maken van Essentiële Modellen, Dietz J. et al., Enterprise Engineering Institute, WmD serie, 2020.

Het basisconcept van de DEMO-methodiek is de *Transactie* waarbij twee *Actoren* door een communicatieproces tot overeenstemmening proberen te komen over een op te leveren of een opgeleverd *Product*. DEMO beschouwt *Transacties* als de fundamentele bouwstenen van een organisatie, die een organisatie op implementatie-onafhankelijke wijze beschrijven.

Een model beschrijft niet alle individuele gevallen afzonderlijk, maar het beschrijft de algemene structuren en wetmatigheden. We spreken in een model dan ook niet van: "*Jan Jansen heeft op 1 jan 2024 een iPhone gekocht*", maar over: "*De persoon doet de verkoop van het product, waarbij de verkoop een verkoopdatum als eigenschap heeft, de persoon een voor- en achternaam als eigenschap heeft, en het product een productnaam als eigenschap heeft*". Een model bestaat dus niet uit specifieke gevallen maar uit abstracte elementen, die de algemene structuur en wetmatigheden beschrijven waaraan die concrete voorbeelden moeten voldoen. Deze abstracte elementen worden klassen genoemd en zijn van een bepaalde soort/type. In DEMO-modellen worden alle basisconcepten weergegeven als klassen van een bepaald soort. Deze basisconcepten van DEMO zijn:

- Een **Transactie:** is een interactie tussen een vragende en uitvoerende partij om met elkaar tot overeenstemming te komen over de levering van een bepaald product. Een *Transactie* is altijd een instantie van één **Transactiesoort**. `Afronden verkoop` en `Afsluiten abonnement` zijn voorbeelden van *Transactiesoorten*.
- Een **Product:** is het (beoogde) resultaat van een succesvol uitgevoerde *Transactie*. Een *Product* is altijd een instantie van één **Productsoort**. Bij elk *Transactiesoort* hoort precies één *Productsoort* en omgekeerd. `Verkoop is afgerond` en `Abonnement is afgesloten` zijn voorbeelden van *Productsoorten*.
- Een **Actor:** is iemand (persoon of groep) die een rol vervult in een *Transactie*. Een *Actor* is altijd een instantie van één **Actorrol**. De *Actorrol* is de "pet" die iemand op heeft. Het duidt de verantwoordelijkheid en bevoegdheid aan die iemand heeft. De *Actorrol* is de bevoegdheid om acties te verrichten die tot zijn verantwoordelijkheid binnen de *Transactie* behoren. Bij een *Transactie* zijn altijd twee *Actoren* betrokken, één als *Initiator* (opdrachtgever) en de ander als *Executor* (uitvoerder).
- Een **Initiator:** is iemand die een *Transactie* start met een verzoek aan de *Executor* om een bepaald *Product* te leveren. Een Initiator is altijd een instantie van één **InitiatorActorrol**. De *Initiator* is bevoegd om een verzoek te doen, om het opgeleverde *Product* te accepteren, om zijn verzoek in te trekken, etc. `Initiator afronden verkoop` en `Initiator afsluiten abonnement` zijn voorbeelden van InitiatorActorrollen.
- Een **Executor:** is iemand die verantwoordelijk is om een bepaald *Product* te leveren. Een *Executor* is altijd een instantie van één **ExecutorActorrol**. De Executor is bevoegd om te beloven om bepaalde verzoeken uit te voeren, om verzoeken te weigeren, etc. `Afronder verkoop` en `Afsluiter abonnement` zijn voorbeelden van een *ExecutorActorrollen*.

DEMO is implementatieonafhankelijk. Het is onafhankelijk van de implementatie van het model in IT-systemen. Maar het is ook onafhankelijk van de implementatie van het model in de organisatie. DEMO definieert geen organisatierollen zoals `Verkoopmedewerker` of `Klant`. DEMO benoemt de individuele verantwoordelijkheden en bevoegdheden, zoals `Verkoop afronder` en `Initiator afronden verkoop`. Hierdoor is het mogelijk om de organisatierollen (bijv. `Verkoopmedewerker`) exact te definiëren door er DEMO *Actorrollen* (`Afronder verkoop` en `Beantwoorder klantvraag`) aan toe te kennen.

DEMO beschouwt de *Transactie* als het fundamentele bouwblok van een organisatie, waarin activiteit, verantwoordelijkheid en informatie op elementair niveau bij elkaar komen. Hierdoor wordt een systematische en samenhangende analyse mogelijk van bedrijfsactiviteiten, bedrijfsinformatie en verantwoordelijkheden en kunnen er beknopte modellen gemaakt worden die inzicht geven in de essentie van een organisatie.

Door het specificeren van typen/soorten (zoals *Transactiesoort* en *Productsoort*) kunnen op type/soort niveau combinaties van soorten worden vastgelegd. Hiermee kan worden afgedwongen dat een bepaald soort *Transactie* (bijvoorbeeld `Afronden verkoop`) altijd resulteert in een bepaald soort *Product* (bijvoorbeeld `Verkoop is afgerond`) en alleen kan worden uitgevoerd door een bepaald soort *ExectorActorrol* (bijvoorbeeld `Afronder verkoop`). Hiervan worden in paragraaf 6.5.1 verdere details weergegeven.

6.3.3 OntoUML

OntoUML is een ontologisch goed onderbouwde uitbreiding van het klassendiagram die voortkomt uit de alom bekende Unified Modeling Language (UML). OntoUML kan worden gebruikt om de informatiestructuren van een domein weer te geven vanuit een implementatie-onafhankelijk perspectief. OntoUML is in de basis niet gericht op het modelleren van de 'activiteiten' die uitgevoerd worden binnen een domein. De DEMO methodiek is daar juist wel op gericht. Daarmee zijn OntoUML en DEMO complementair aan elkaar en worden

in combinatie gebruikt om de formele aspecten van een domein vast te leggen. OntoUML-elementen worden weergegeven als UML-klassen en associaties voorzien van stereotypen (tags boven de naam van het element, voorzien van dubbele haakjes « »), waarbij het stereotype de ontologische eigenschappen/betekenis achter de klasse of associatie aangeeft. In deze paragraaf leggen we de meest relevante ontologische eigenschappen en stereotypen uit voor de ontwikkeling van dit document.

Een belangrijk ontologisch eigenschap van klassen in OntoUML is **rigiditeit**:

- Een **rigide** klasse classificeert statisch zijn instanties, d.w.z. een instantie van een rigide klasse is gedurende zijn hele bestaan een instantie van een rigide klasse. Voorbeelden hiervan zijn: `Persoon`, `Dier` en `Organisatie`.
- Een **anti-rigide** klasse classificeert dynamisch zijn instanties, d.w.z. iets kan op een bepaald tijdstip een instantie van een anti-rigide klasse zijn, maar op een later tijdstip niet meer. Voorbeelden hiervan zijn: `Tiener`, `Volwassene`, `Student` en `Echtgenoot`.

Een andere fundamenteel ontologisch eigenschap van klassen is **sortaliteit**:

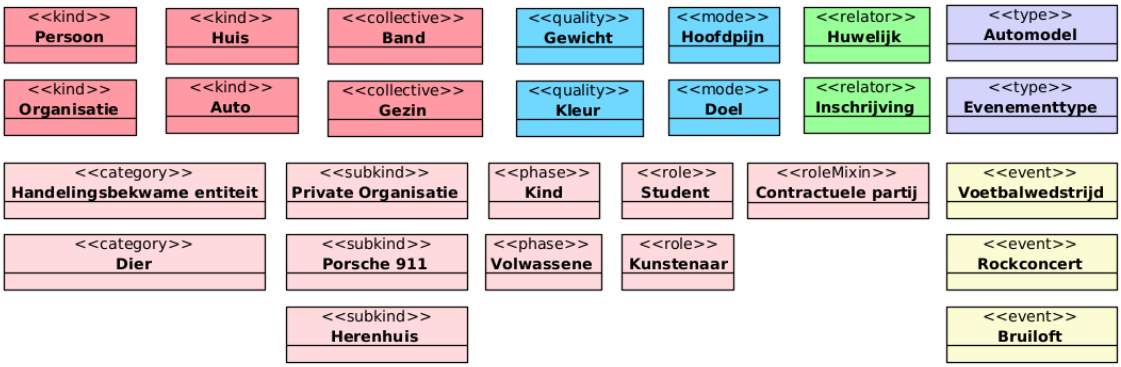
- Een **sortale** klasse is een klasse waarvan de instanties een gemeenschappelijk identiteitsprincipe delen, waarbij de sortale klasse dit principe verschaft of erft van een superklasse. Voorbeelden van klassen die identiteit geven zijn: `Persoon`, `Huis` en `Auto`. Voorbeelden van klassen die identiteit erven zijn: `Student`, `Man`, `Volwassene`, `Herenhuis` en `Sportwagen`.
- Een **niet-sortale** klasse is een klasse waarvan de instanties entiteiten bevat die voldoen aan verschillende identiteitsprincipes. Voorbeelden hiervan zijn: `Handelingsbekwame entiteit`, die instanties van sortale klassen `Person` en `Organisatie` omvat; en `Fysiek object`, dat o.a. instanties van sortale klassen `Auto` en `Huis` omvat.

De stereotypen van de OntoUML-klasse die in dit document worden gebruikt, zijn:

- **«kind»** is een rigide, sortale (identiteit verschaffende) klasse die objectachtige entiteiten classificeert. Voorbeelden hiervan zijn: `Persoon`, `Organisatie`, `Auto` en `Huis`.
- **«collective»** is een rigide, sortale klasse en wordt gebruikt om concepten weer te geven die een homogene interne structuur hebben, waarbij alle delen door het geheel op dezelfde manier worden waargenomen. Voorbeelden hiervan zijn: `Band` als «collective» die bestaat uit `Bandleden`; `Gezin` als «collective» die bestaat uit `Gezinsleden`.
- **«quality»** is een rigide, sortale (identiteit verschaffende) klasse die entiteiten classificeert die aspecten van andere entiteiten vertegenwoordigen en meetbaar zijn in een bepaalde waarderuimte (d.w.z. conceptuele ruimte). Een «quality» kan zijn gebruikt om individuen te vergelijken, op basis van de waarde die het inneemt in een bepaalde kwaliteitsruimte (bijvoorbeeld, een massa in de kilogramschaal, of een positie binnen het RGB-spectrum). Voorbeelden hiervan zijn: `Gewicht` (zoals in het gewicht van een persoon), `Naam` (zoals in de naam van een organisatie), `Kleur` (zoals in de kleur van een auto), en `Duur` (zoals in de duur van een concert).
- **«mode»** is een bepaald type intrinsieke eigenschap die geen gestructureerde waarde heeft. Net als kwaliteiten zijn modi ook individuen die existentieel afhankelijk zijn van hun dragers. Typen die worden gestereotypeerd als «mode» zijn ook rigide. Een voorbeeld hiervan is: `Hoofdpijn` (klasse met stereotype «mode») als eigenschap van een `Persoon` (klasse met stereotype «kind»), of `Doel` als eigenschap van een `Organisatie`.
- **«relator»** is een rigide, sortale (identiteit verschaffende) klasse die relationele entiteiten classificeert, ook wel bekend als relaties. Voorbeelden hiervan zijn: `Huwelijk`, dat betrekking heeft op twee gevallen van `Echtgenoot`; `Contract` waarin meerdere gevallen van `Partij` in het kader van formele

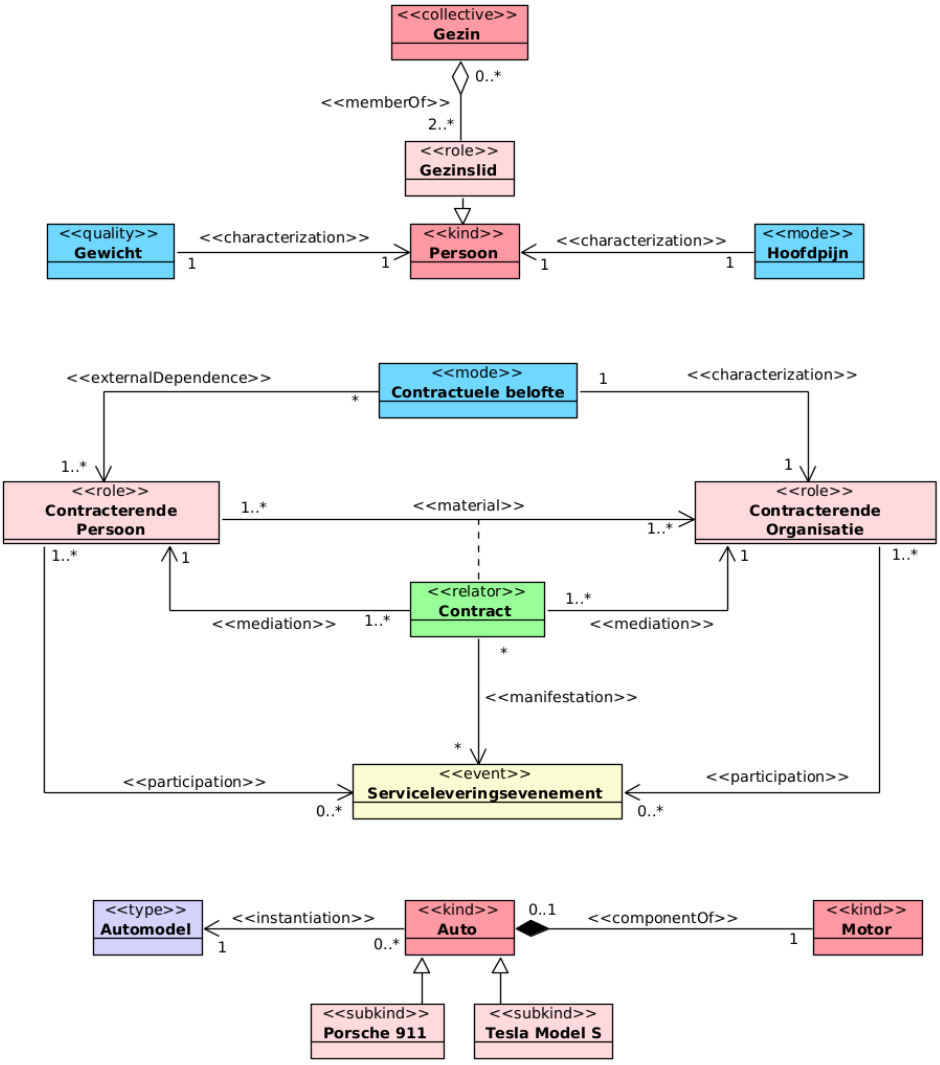
overeenkomsten worden vermeld; en `Inschrijving`, die relateert een instantie van `Student` aan een instantie van `Onderwijsinstelling`.

- **«category»** is een rigide niet-sortale klasse die entiteiten van verschillende soorten classificeert. Voorbeelden hiervan zijn `Handelingsbekwame entiteit`, die instanties van zowel `Persoon` als `Organisatie` classificeert; en `Dier` die gevallen van verschillende diersoorten classificeert, waaronder `Felis Catus` (huiskat), `Canis Lupus Familiaris` (hond) en `Panthera Leo` (leeuw).
- **«subkind»** is een rigide sortale klasse die zijn identiteit erft van een andere sortal. Voorbeelden hiervan zijn: `Publieke organisatie`, `Porsche 911` en `Herenhuis`.
- **«phase»** is een anti-rigide sortale klasse waarvan de instantiatie afhangt van een verandering in een intrinsieke eigenschap. `Kind` kan bijvoorbeeld een subklasse zijn van `Persoon` wiens instanties mensen tot 12 jaar oud zijn. `Volwassene` is een subklasse van `Persoon` wiens instanties mensen van 18 jaar en ouder zijn.
- **«role»** is een anti-rigide sortale klasse waarvan de instantiatie afhangt van een relationele voorwaarde. Voorbeelden hiervan zijn: `Student` en `Kunstenaar`. Een `Student` kan bijvoorbeeld worden gedefinieerd als een instantie van `Persoon` in relatie tot `Schoolinschrijving`. Alleen een `Persoon` kan de rol van `Student` vervullen.
- **«roleMixin»** is een anti-rigide niet-sortale klasse die rollen classificeert die speelbaar zijn door individuen van verschillende soorten. Voorbeelden hiervan zijn: De rol van `Contractuele partij` welke kan worden gedefinieerd als een `Handelingsbekwame entiteit` (zoals `Persoon` of `Organisatie`) die een rol speelt in een `Contract`. Dus zowel een `Persoon` als een `Organisatie` kan de rol van `Contractuele partij` vervullen.
- **«type»** is een rigide klasse die entiteiten classificeert die zelf instanties hebben (d.w.z. andere klassen). Voorbeelden hiervan zijn: het `Automodel`, waarvan de instanties `Porsche 911` en `Tesla Model S` kunnen zijn; en `Evenementtype`, waarvan de instanties `Muziekevenement` en `Sportevenement` kunnen omvatten.
- **«event»** identificeert die klassen waarvan de instanties gebeurtenissen zijn (gebeurtenissen in het verleden). «event» klassen staan los van alle klassen die duurzame typen modelleren waarvan de instanties blijvend zijn. Als gevolg hiervan mag geen enkele klasse worden gestereotypeerd met of een specialisatie zijn van zowel «event» als een van de andere stereotypen die zijn gedefinieerd voor duurzame typen in OntoUML. Klassen met stereotype «event» kunnen speciale attributen krijgen om de temporele eigenschappen (tijdsaspecten als begin en eind) van een gebeurtenis weer te geven. Voorbeelden van «event» klassen zijn: `Voetbalwedstrijd`, `Rockconcert` en `Bruiloft`.

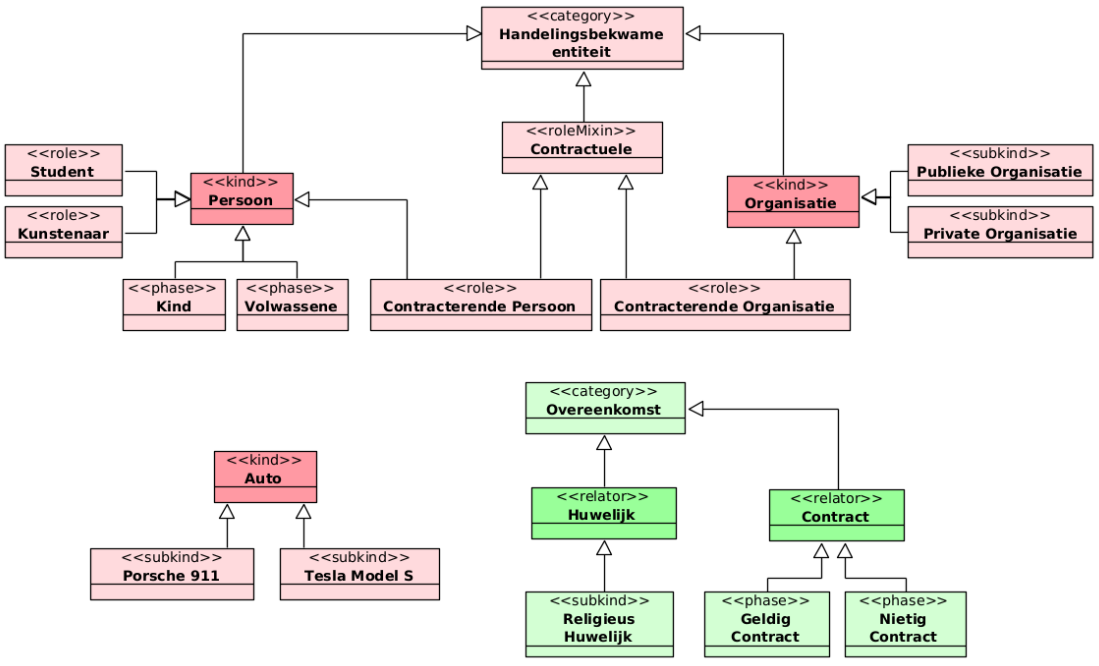


Figuur 11: Voorbeelden van UML-klassen voorzien van OntoUML-stereotypen

Figuur 11 geeft een opsomming van een aantal voorbeelden van UML-klassen die voorzien zijn van OntoUML-stereotypen. In Figuur 12 is voor een aantal voorbeelden van UML-klassen de generalisatie relatie toegevoegd.



Figuur 12: Voorbeelden van UML-klassen voorzien van OntoUML-stereotypen inclusief generalisaties



Figuur 13: Voorbeelden van UML-associaties voorzien van OntoUML-stereotypen

De stereotypen van de OntoUML-associatie die in dit document worden gebruikt, zijn:

- **«mediation»** is een existentiële afhankelijkheidsrelatie die een «relator» verbindt met de entiteiten die de «relator» relateert. Voorbeelden hiervan zijn: een instantie van de «relator» `Huwelijk` wordt middels «mediation» verbonden met twee instanties van de klasse `Echtgenoot`; een instantie van de «relator» `Contract` wordt middels «mediation» verbonden aan twee of meer instanties van de klasse `Contractpartij`.
- **«characterization»** is een existentiële afhankelijkheidsrelatie die een aspect (bijv. een kwaliteit «quality» or een intrinsieke eigenschap) verbindt met de entiteit die het kenmerkt. Voorbeelden hiervan zijn: instanties van de klasse `Kleur` karakteriseren bijvoorbeeld instanties van de klasse `Fysiek object`.
- **«externalDependence»** is een relatie die aangeeft dat de ene instantie van een klasse extern afhankelijk is van een instantie van een andere klasse. Deze soort relatie is toepasbaar op in een sociale interactie tussen twee Handelingsbekwame entiteiten, waarbij de entiteit A een sociale toezegging doen naar entiteit B. In die situatie is entiteit B extern afhankelijk van de toezegging gedaan door entiteit A. Een voorbeeld hiervan is: Een `Organisatie` (klasse met stereotype «kind») is afhankelijk (associatie met stereotype «externalDependence») van de `Belofte tot betalen` (klasse met stereotype «mode») die een `Persoon` (klasse met stereotype «kind») doet.
- **«material»** is een relatie die entiteiten verbindt op basis van iets dat afhankelijk is van deze entiteiten. Voorbeelden hiervan zijn: instanties van de associatie `getrouwd met` verbinden instanties van de klasse `Echtgenoot` die een huwelijksrelatie hebben die van hen afhankelijk is.
- **«componentOf»** is een deel-geheel relatie die objecten verbindt met andere objecten die daar onderdeel van uit maken. Voorbeelden hiervan zijn: de compositierelatie tussen een instantie van `Auto` en de instantie van `Motor` die daar onderdeel van uitmaakt; de relatie tussen een instantie van `Menselijk lichaam` en een instantie van `Hart`.
- **«memberOf»** is een deelrelatie tussen een functioneel complex of een «collective» (als deel) en een «collective» (als geheel). Voorbeelden hiervan zijn: Een `Bandlid` is lid van («memberOf») een `Band`.
- **«historicalDependence»** is een relatie die entiteiten bindt vanwege een gebeurtenis die plaatsvond in de verleden tijd. Een voorbeeld hiervan is: een instantie van `Plaats` kan via een historische afhankelijkheid in verband worden gebracht met een instantie van `Foto` dat een weergave is van die plaats op een bepaald moment in de tijd.
- **«instantiation»** is een relatie tussen twee klassen die voorstelt dat instanties van één kunnen zijn geclassificeerd door instanties van de andere. Een voorbeeld hiervan is: de relatie tussen `Auto` en `Model`, die geeft aan dat elke auto een exemplaar is van een model.
- **«manifestation»** is een relatie tussen een duurzame klasse en gebeurtenis (een klasse met stereotype «event») die aangeeft dat de gebeurtenis een manifestatie is van die duurzame klasse. Een voorbeeld hiervan is: Een `Bruiloft` (klasse met stereotype «event») is de manifestatie van de eigenschappen van `Huwelijk` als klasse met stereotype «relator».
- **«participation»** is een relatie die de deelname van duurzame klassen aan gebeurtenissen (klassen met stereotype «event») aangeeft. Een voorbeeld hiervan is: Een `Persoon` (klasse met stereotype «kind») neemt deel aan (associatie met stereotype «participation») een `Handeling van componeren` (klasse met stereotype «event»).
- **«creation»** is een relatie die aangeeft dat een duurzame klasse wordt gecreëerd tijdens een gebeurtenis (klasse met stereotype «event»). Een voorbeeld hiervan is: een `Muziekstuk` (klasse met stereotype «kind») wordt gecreëerd (middels de associatie met stereotype «creation») in de gebeurtenis `Handeling van componeren` (klasse met stereotype «event»).
- **Geen stereotype:** associaties die niet kunnen worden getypeerd door een OntoUML stereotype mogen zonder stereotype worden weergegeven.

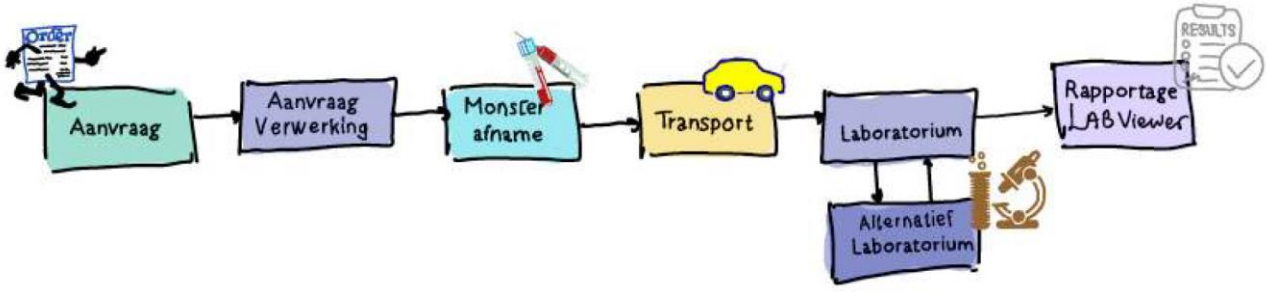
In Figuur 13 is een aantal voorbeelden van associaties tussen UML-klassen weergegeven voorzien van OntoUML-stereotypen.

6.4 Activiteiten beschrijvend

Deze paragraaf bevat de informele beschrijving van de 'activiteiten' die plaatsvinden in het Diagnostiek sub-domein weergegeven in de ArchiMate modelleringstaal. Hierbij wordt niet beschreven HOE deze activiteiten plaatsvinden, maar wordt alleen beschreven WAT binnen het Diagnostiek sub-domein aan activiteiten wordt uitgevoerd.

6.4.1 Scope en disclaimer

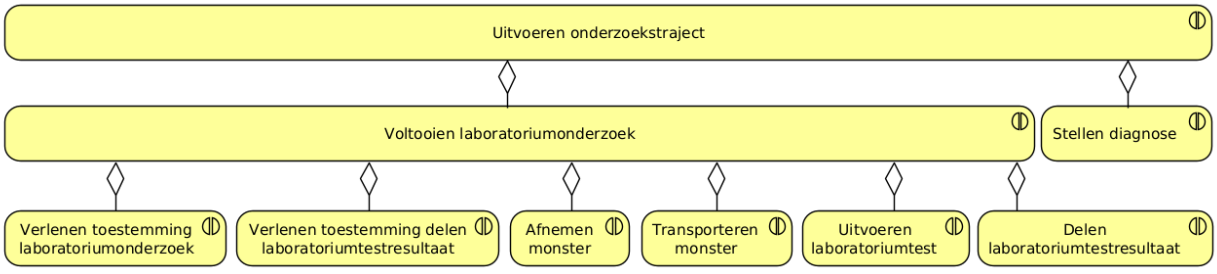
De modellen in deze paragraaf zijn gebaseerd op een analyse van het document: "IHE Handreiking – Digitale uitwisseling laboratoriumgegevens – Vormgegeven op basis van nationale en internationale standaarden, Versie 1.0 2022." De modellen zijn geen volledige weergave van de inhoud van de IHE Handreiking (zie Figuur 14). Ook is er een aantal toevoegingen gedaan en wijzigingen doorgevoerd, ter herkennen aan onder andere de afwijkende naamgeving in paragraaf 6.4.2 ten opzichte van Figuur 14. Het doel van paragraaf 6.4.2 is om een beknopte, samenhangende voorbeeldcasus te kunnen presenteren in beschrijvende vorm, om voldoende inhoud te bieden voor de formele uitwerking in paragraaf 6.7. Het doel van deze formele uitwerking is om een duidelijk beeld geven van de principes, de samenhang en het uiterlijk van het op te leveren IZB-domeinmodel. Het doel van dit document is niet om het definitieve, complete en correcte IZB-domeinmodel te presenteren. De domeinmodelleringsmethode is nog in ontwikkeling en dit domeinmodel is daarom nog niet correct en nog niet volledig.



Figuur 14: Transmurale laboratorium proces (bron: IHE handreiking)

6.4.2 Procesbouwblokken

Figuur 15 laat zien uit welke procesbouwblokken het sub-domein Diagnostiek bestaat. In de rest van deze paragraaf worden de individuele procesbouwblokken kort toegelicht. De teksten gemarkeerd met * zijn overgenomen uit de IHE Handreiking. Afwijkingen in de tekst ten opzichte van de gebruikte bron (IHE Handreiking) zijn gemarkeerd met: {<toegevoegde tekst>} of {…} op de plek waar tekst is verwijderd.



Figuur 15: Procesbouwblokken weergegeven als Bedrijfsinteracties in ArchiMate

6.4.2.1 *Uitvoeren onderzoekstraject*

Dit procesbouwblok leidt tot het uitvoeren van een onderzoekstraject. De exacte details van wat het 'Uitvoeren onderzoekstraject' procesbouwblok precies inhoudt zijn op het moment van schrijven nog niet volledig uitgewerkt.

6.4.2.2 *Stellen diagnose*

Dit procesbouwblok leidt tot het stellen van een diagnose op basis van het laboratoriumtestresultaat. De exacte details van wat het 'Stellen diagnose' procesbouwblok precies inhoudt zijn op het moment van schrijven nog niet volledig uitgewerkt.

6.4.2.3 *Voltooien laboratoriumonderzoek**

{Dit procesbouwblok} start bij de partij die een aanvraag doet bij een laboratorium (zorgaanbieder). De aanvragende partij kan een zorgverlener zijn of de patiënt/cliënt. De aanvraagverwerker ontvangt {…} de aanvraag. Soms op papier, maar uiteindelijk bij voorkeur digitaal. Deze aanvraag bevat alle gegevens die nodig zijn voor het vervolg {…}. Denk hierbij aan de persoonsgegevens van de patiënt, gegevens van de aanvrager, eventuele informatie over wie de uitslag (rapportage) moet ontvangen en natuurlijk de onderzoeksvraag in de vorm van de uit te voeren testen. De taak van {de aanvraagverwerker} is om de gegevens uit de aanvraag te controleren en eventueel aan te vullen. Denk hierbij aan: onvolledige informatie over de patiënt, een onbekende aanvrager, de uit te voeren onderzoeken is niet helder. Daarnaast moet gecontroleerd worden of de uit te voeren onderzoeken valide zijn voor deze patiënt en aanvrager.

6.4.2.4 *Verlenen toestemming laboratoriumonderzoek*

Dit procesbouwblok leidt tot het verkrijgen van toestemming voor het uitvoeren van een laboratoriumonderzoek op een persoon. De exacte details van wat het 'Verlenen toestemming laboratoriumonderzoek' procesbouwblok precies inhoudt zijn op het moment van schrijven nog niet volledig uitgewerkt.

6.4.2.5 *Verlenen toestemming delen laboratoriumtestresultaat*

Dit procesbouwblok leidt tot het verkrijgen van toestemming voor het delen van de laboratoriumuitslag met derden. De exacte details van wat het 'Verlenen toestemming delen laboratoriumtestresultaat' procesbouwblok precies inhoudt zijn op het moment van schrijven nog niet volledig uitgewerkt.

6.4.2.6 *Afnemen monster**

Vervolgens volgt afname van het monster, doorgaans door een medewerker van het laboratorium. Dit gebeurt bij het laboratorium zelf, een prikpost van het laboratorium, een verpleegafdeling, bij de GGD of bij de patiënt thuis. Tijdens het monster {afnemen} wordt het daadwerkelijke monster bij de patiënt afgenomen en aan de aanvraag gekoppeld. {De monsterafnemer} is verantwoordelijk voor het labelen van het monster (één of meerdere), het correct koppelen van het monster aan de aanvraag en met zekerheid vaststellen dat het monster ook van de patiënt afkomstig is. Om dit te kunnen uitvoeren {is} informatie nodig over het af te nemen monster en identificatie informatie van de patiënt. {Tevens} worden gegevens over het monster en de afname vastgelegd. Denk hierbij bijvoorbeeld aan tijdstip, locatie en persoon die het monster heeft afgenomen.

6.4.2.7 *Transporteren monster**

Het monster wordt dan met de relevante informatie getransporteerd naar het laboratorium. Het transport wordt verzorgd door het laboratorium zelf (bijvoorbeeld door de medewerker die het monster heeft afgenomen) of door een partij die de monsters ophaalt en bezorgt bij het laboratorium.

6.4.2.8 *Uitvoeren laboratoriumtest**

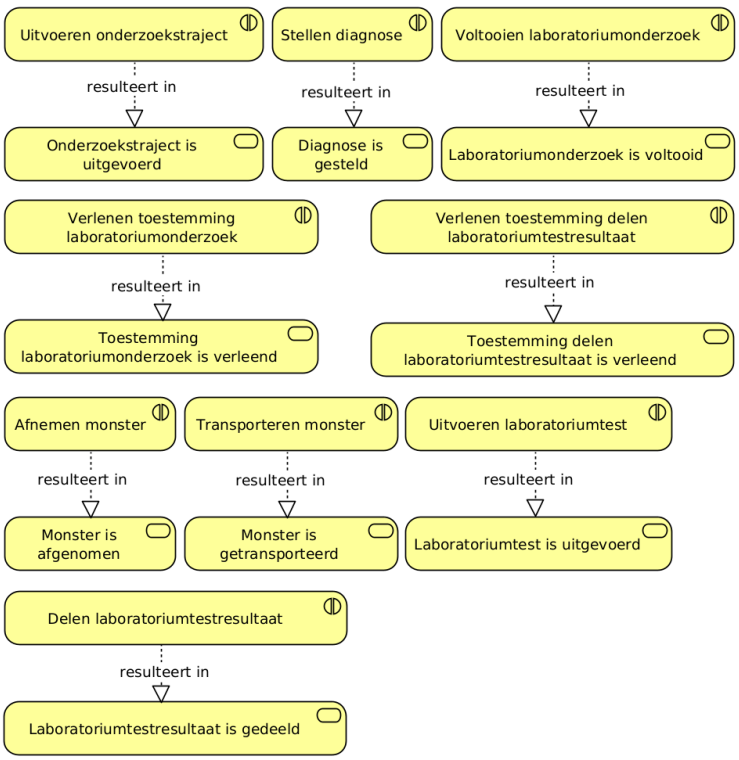
Het resultaat {tot nu toe} is een gecontroleerde aanvraag met de bijbehorende monsters. Tijdens deze {activiteit} zal de aanvraag eventueel gesplitst worden en de individuele of groepen testen uitgevoerd worden. Deze groepen kunnen in één of meerdere laboratoria worden uitgevoerd. {Dit procesbouwblok} is verantwoordelijk voor het correct uitvoeren van de testen die in de aanvraag genoemd zijn. Daarnaast zal {dit procesbouwblok} de resultaten autoriseren. De bepaling wordt uitgevoerd door het laboratorium waar de aanvraag is geplaatst. In het geval dat het betreffende laboratorium de bepaling niet kan doen wordt de uitvoering uitbesteed aan een ander laboratorium. In de praktijk blijkt dat slechts een klein deel van de aanvragen worden uitbesteed.

6.4.2.9 *Delen laboratoriumtestresultaat**

De aanvraag bevat de gegevens van de aanvrager en eventuele kopie aanvragers. Het systeem {…} zorgt voor de juiste rapportage met de zoals in afspraken, onder andere in de aanvraag, vastgelegde partijen. Deze {activiteit} wordt uitgevoerd door verschillende partijen, vrijwel altijd de partij die ook de aanvraag heeft gedaan. In de praktijk worden er echter ook vaak afschriften van (delen van) de resultaten naar derde partijen gestuurd zoals apothekers. Voor sommige partijen geldt een wettelijke plicht in het leveren van deze gegevens. Voor andere zijn er lokale afspraken gemaakt. Meer en meer is er ook een rapportage direct naar de patiënt (wettelijk recht) welke meer duiding nodig heeft. Vanuit één set aan gegevens kunnen er meerdere rapporten gestuurd worden. Het zal duidelijk zijn dat in dit kader privacywetgeving van toepassing is.

6.4.3 Producten

Conform de specificatie zoals beschreven in paragraaf 6.3.2, leidt elk procesbouwblok tot één *Product*. Figuur 16 laat zien welke producten door welke procesbouwblokken (zoals weergegeven in Figuur 15) worden gerealiseerd.



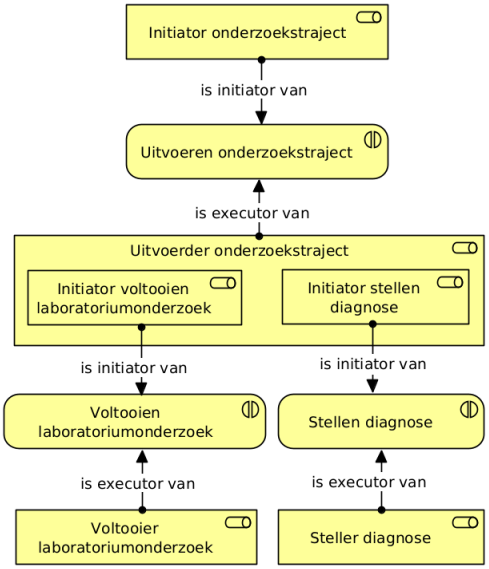
Figuur 16: Producten die gerealiseerd worden door procesbouwblokken weergegeven in ArchiMate

6.4.4 Coördinatiestructuren

In de rest van paragraaf 6.4 brengen we de in paragraaf 6.4.2 beschreven individuele procesbouwblokken bij elkaar door de achterliggende coördinatiestructuur weer te geven.

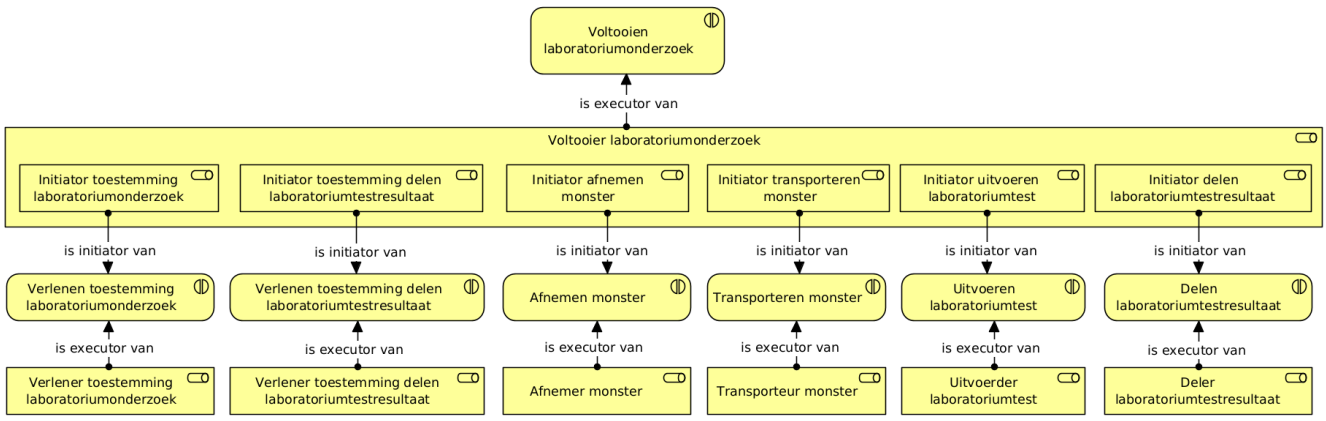
6.4.4.1 Voltooien onderzoekstraject en diagnose

Zoals in Figuur 17 is weergegeven, staat in de totale coördinatiestructuur van het sub-domein Diagnostiek het procesbouwblok `Uitvoeren onderzoekstraject` aan de oorsprong. Vanuit dit procesbouwblok kunnen twee andere bouwblokken worden gecoördineerd, zijnde `Voltooien laboratoriumonderzoek` en `Stellen diagnose`. Daarmee wil niet gezegd worden dat deze twee procesbouwblokken altijd alleen vanuit het procesbouwblok `Voltooien onderzoekstraject` kunnen worden geïnitieerd. `Voltooien laboratoriumonderzoek` kan ook vanuit een andere context worden opgestart. Dit geldt ook voor `Stellen diagnose`, alhoewel dat procesbouwblok wel afhankelijk is van externe informatie op basis waarvan een diagnose kan worden vastgesteld. Verdere details hiervan zijn terug te vinden in de formele specificaties van het sub-domein Diagnostiek in paragraaf 6.7.



Figuur 17: Coördinatie Structuur Diagram voor 'voltooien onderzoekstraject' en 'voltooien diagnose' procesbouwblokken weergegeven in ArchiMate

6.4.4.2 Voltooien laboratoriumonderzoek



Figuur 18: Coördinatie Structuur Diagram voor 'voltooien laboratoriumonderzoek' en onderliggende procesbouwblokken weergegeven in ArchiMate

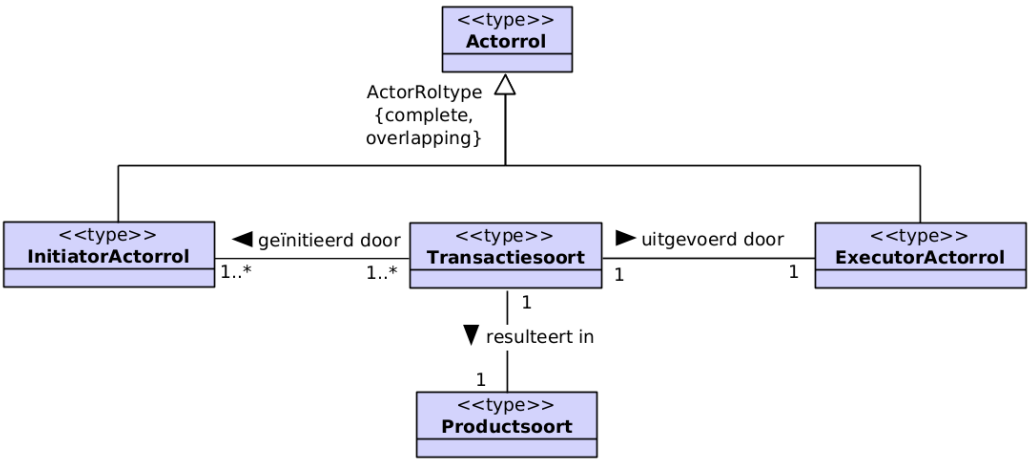
Figuur 18 geeft de volledige coördinatiestructuur weer die onderliggend is aan het procesbouwblok `Voltooien laboratoriumonderzoek`. Hierin is te zien dat de realisatie van dit procesbouwblok bestaat uit de coördinatie van 6 onderliggende procesbouwblokken, waarbij de *Actor* die de rol van *Executor* speelt in `Voltooien laboratoriumonderzoek` ook de *Actor* is die de rol van *Initiator* speelt voor de 6 onderliggende procesbouwblokken.

6.5 Formeel metamodel voor Activiteiten

Een domeinmodel geeft een beschrijving van een domein én van zichzelf. Dat laatste betekent dat het model van het domeinmodel ook nauwkeurig is beschreven. Dit is wat ook wel het metamodel van een domeinmodel wordt genoemd. In feite vormt dit metamodel de leeswijzer voor het domeinmodel. Zonder deze leeswijzer is een domeinmodel niet of moeilijk te interpreteren. Met name in het formeel specificeren van een domeinmodel is het essentieel dat dit niet verkeerd geïnterpreteerd kan worden. Als een leverancier van te realiseren software een domeinmodel verkeerd kan interpreteren, is er immers een grotere kans dat de gerealiseerde software niet voldoet aan de eisen en wensen van de klant. Daarom wordt in paragraaf 6.5 het metamodel van de in paragraaf 6.7 beschreven formele specificaties van het voorbeeld domeinmodel toegelicht. Dit is uitgewerkt in de vorm van een aantal metamodellen. Net zoals het domeinmodel zelf, is het metamodel in feite een combinatie van een aantal modellen. Zoals eerder aangegeven, wordt in het formeel specificeren van het domeinmodel gebruik gemaakt van een combinatie van twee talen/methodieken: OntoUML en DEMO. De metamodellen zijn weergegeven in de OntoUML-taal, maar de structuur en betekenis van de 'activiteiten' onderdelen in de metamodellen zijn mede gebaseerd op de DEMO-methodiek. Er is geen apart metamodel voor het 'informatie' onderdeel van het domeinmodel beschreven in deze paragraaf, omdat hiervoor de OntoUML-taal wordt gebruikt, welke in paragraaf 6.3.3 al is beschreven.

6.5.1 Typen/soorten

Net zoals standaard UML, maakt OntoUML een onderscheid tussen klassen en instanties van klassen. OntoUML kent echter ook een hoger gelegen typering van klassen in de vorm van het «type» stereotype. Zoals beschreven in paragraaf 6.3.3, is een klasse van het «type» stereotype een rigide klasse is die entiteiten classificeert die zelf instanties hebben (d.w.z. andere klassen). De DEMO-methodiek kent ook het concept van soorten/typen, zoals ook aangegeven in paragraaf 6.3.2. Figuur 19 geeft weer hoe de soorten/typen uit DEMO in de OntoUML taal zijn vastgelegd in het metamodel.



Figuur 19: Typen en soorten uit DEMO weergegeven in OntoUML

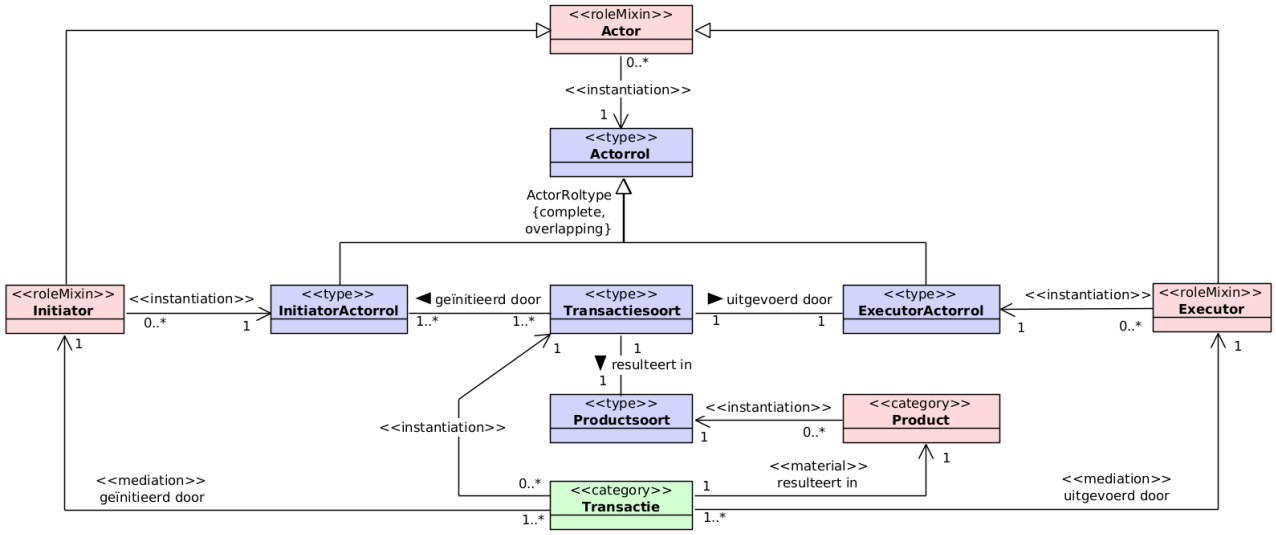
Door op hoger geleden typeniveau de associaties tussen type/soorten te specificeren, worden in Figuur 19 beperkingen aangebracht in de combinaties van actorrollen, transactiesoorten en productsoorten:

- Een *Transactiesoort* wordt altijd geïnitieerd door één *InitiatorActorrol*; een *InitiatorActorrol* kan wel meerdere *Transactiesoorten* initiëren.
- Een *Transactiesoort* wordt altijd uitgevoerd door één *ExecutorActorrol*; een *ExecutorActorrol* kan ook maar één *Transactiesoort* uitvoeren.
- Een *Transactiesoort* resulteert in altijd één *Productsoort*; een *Productsoort* kan worden gerealiseerd door maar één *Transactiesoort*.

In Figuur 19 is ook door middel van de 'ActorRoltype' beperking bepaald dat *Actorrol* maximaal twee specialisaties kent in de vorm van *InitiatorActorrol* en *ExecutorActorrol* (aangegeven door 'complete'). Diezelfde beperking geeft (aangegeven door 'overlapping') aan dat sommige instanties van *Transactiesoort* gerelateerd zijn met maar één instantie van «type» *Actorrol*.

6.5.2 Instanties van typen en soorten

Elk van de soorten en typen die zijn weergegeven in Figuur 19 kan worden geïnstantieerd naar klassen welke in Figuur 20 zijn weergegeven. Van alle klassen weergegeven in Figuur 20 is in paragraaf 6.3.2 een definitie gegeven.



Figuur 20: Instanties van typen en soorten uit Figuur 19

Wat Figuur 20 toevoegt aan het metamodel is dat:

- Van alle «type» klassen in het metamodel (zijnde *Actorrol*, *InitiatorActorrol*, *Executoractorrol*, *Transactiesoort* en *Productsoort*) nul tot meerdere instanties kunnen bestaan.
- De klassen *Actor*, *Initiator*, *Executor, Transactie* en *Product* altijd een «instantiation» zijn van één type.
- Een *Transactie* altijd wordt *geïnitieerd door* één *Intiator*; een *Initiator* één of meerdere *Transacties* kan initiëren.
- Een *Transactie* altijd wordt *uitgevoerd door* één *Executor*; een *Executor* één of meerdere *Transacties* kan uitvoeren.
- Een *Transactie* resulteert in precies één *Product*; een *Product* altijd het resultaat is van één *Transactie*.

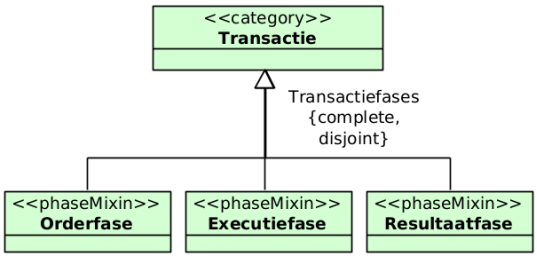
6.5.3 Transactiefases

Een *Transactie* kan zich in één van drie fases bevinden (zie Figuur 21):

- **Orderfase:** is de fase van een *Transactie* waarin de *Initiator* en de *Executor* trachten consensus te bereiken over het door de *Executor* voort te brengen *Product*. De *Orderfase* begint met het doen van het *Verzoek* (*Coördinatie-actie*) door de *Initiator* en eindigt succesvol met de *Belofte* (*Coördinatie-actie*) door de *Executor*.

- **Executiefase:** is de fase van een *Transactie* waarin de *Executor* het *Product* van de *Transactie* tot stand brengt door het uitvoeren van de *Productie-actie*.
- **Resultaatfase:** is de fase van een *Transactie* waarin de *Initiator* en de *Executor* trachten consensus te bereiken over het door de *Executor* tot stand gebrachte *Product*. De *Resultaatfase* begint met de *Verklaring* (*Coördinatie-actie*) door de *Executor* dat het *Product* tot stand is gebracht en eindigt succesvol met de *Aanvaarding* (Coördinatie-actie) daarvan door de *Initiator*.

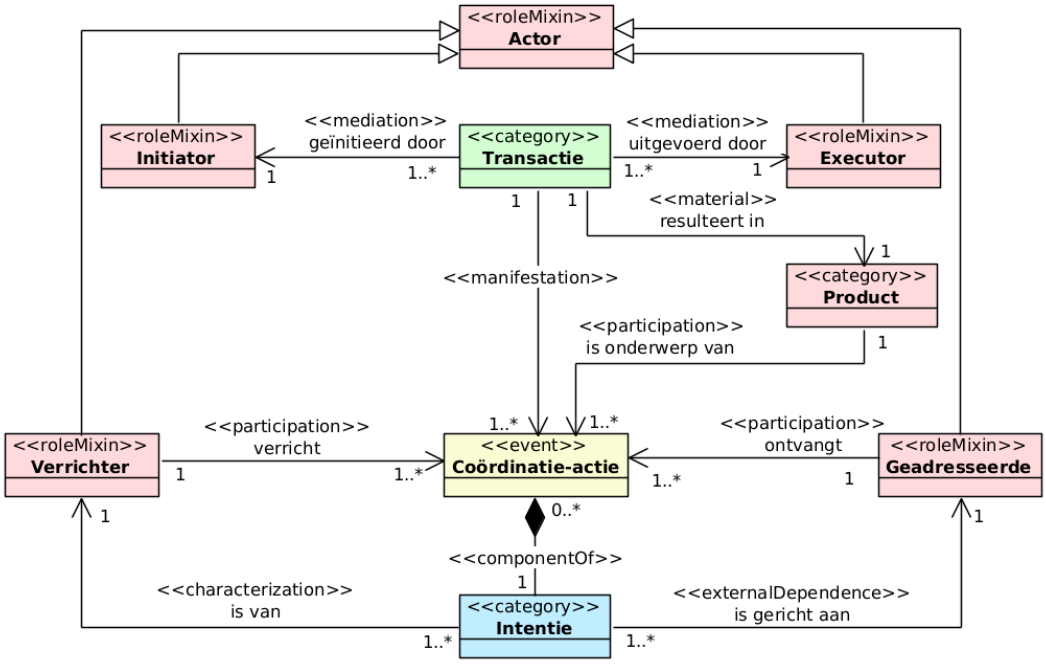
Omdat een *Transactie* zich op elk moment in de tijd maar in één fase kan bevinden, is een beperking 'Transactiefase 'toegevoegd (zie Figuur 21) die 'disjoint 'aangeeft (i.p.v. 'overlapping)'. Er zijn maximaal drie fases waarin een transactie zich kan bevinden. Vandaar is de beperking 'complete'.



Figuur 21: De drie fases van een transactie

6.5.4 Het metamodel van een transactie (Coördinatie-perspectief)

Gedurende de *Orderfase* en de *Resultaatfase* van een *Transactie* vindt coördinatie tussen de *Initiator* en de *Executor* plaats. Figuur 22 geeft het metamodel van een *Transactie* vanuit dit coördinatie-perspectief weer.



Figuur 22: Ontologie van een transactie vanuit het Coördinatie-perspectief

Ten opzichte van Figuur 20 introduceert Figuur 22 een aantal nieuwe begrippen die te maken hebben met deze coördinatie tussen *Initiator* en *Executor*:

- **Coördinatie-actie** is de atomaire eenheid van activiteit in een organisatie. Een *Coördinatie-actie* heeft vier kenmerken: de *Verrichter*, de *Intentie*, de *Geadresseerde* en het *Onafhankelijk productiefeit*. Een voorbeeld hiervan is: `Initiator afronden verkoop (Verrichter) verzoekt`

(Intentie) `Verkoop afronder` *(Geadresseerde)* `<Afronden verkoop> is voltooid` *(Onafhankelijk productiefeit)*.

- **Verrichter** is de *Actor* die een *Coördinatie-actie* verricht. Zowel de *Initiator* als de *Executor* van een *Transactie* kunnen als *Verrichter* van een *Coördinatie-actie* optreden. Een voorbeeld hiervan is: `Initiator afronden verkoop.`
- **Intentie** is de 'sociale' positie die de *Verrichter* van een *Coördinatie-actie* inneemt tegenover de *Geadresseerde*, met betrekking tot het *Product* van de *Transactie*. Voorbeelden hiervan zijn: `Verzoek`, `Belofte`, `Verklaring` en `Aanvaarding`. Zie paragraaf 6.5.5 van alle soorten intenties.
- **Geadresseerde** is de *actor* tot wie een *Coördinatie-actie* is gericht. Zowel de *Initiator* als de *Executor* van een *Transactie* kunnen als *Geadresseerde* van een *Coördinatie-actie* optreden. Een voorbeeld hiervan is: `Verkoop afronder`.

Wat Figuur 22 toevoegt aan het metamodel is dat:

- Een *Transactie* wordt gemanifesteerd door één of meerdere *Coördinatie-acties*; een *Coördinatie-actie* altijd een manifestatie is van één *Transactie*.
- Een *Coördinatie-actie* altijd door één *Verrichter* wordt verricht; een *Verrichter* één of meerdere *Coördinatie-acties* kan verrichten.
- Een *Coördinatie-actie* altijd door één *Geadresseerd* wordt ontvangen; een *Geadresseerde* één of meerdere *Coördinatie-acties* kan ontvangen.
- Een *Coördinatie-actie* altijd bestaat uit één *Intentie*; een *Intentie* mogelijk in één of meerdere *Coördinatie-acties* kan voorkomen.
- Een *Verrichter* wordt gekarakteriseerd door één of meerdere *Intenties*; een *Intentie* als onderdeel van een *Coördinatie-actie* altijd een karakterisering van één *Verrichter* is.
- Een *Geadresseerde* extern afhankelijk is van één of meerdere *Intenties*; een *Intentie* als onderdeel van een *Coördinatie-actie* altijd aan één *Geadresseerde* gericht is.

Wat opvalt in Figuur 22 is dat twee wezenlijke concepten uit de DEMO-taal ontbreken, namelijk:

- **Onafhankelijk productiefeit** is een zin die een *Propositie* uitdrukt betreffende de *Geproduceerde entiteit* als resultaat van een *Transactie*. Een voorbeeld hiervan is: `Afronden verkoop` *(Geproduceerde entiteit)* `is voltooid` *(Propositie)*.
- **Coördinatiefeit** is het resultaat van een succesvol verrichte *Coördinatie-actie*. Een *Coördinatiefeit* bevat alle gegevens die over de uitgevoerde *Coördinatie-actie* inclusief de *Verrichter*, *Intentie*, *Geadresseerde* en *Onafhankelijk productiefeit*, en het moment (datum en tijd) waarop de *Coördinatie-actie* is verricht.

Er is bewust gekozen voor het niet opnemen van deze twee feittypen wegens het wezenlijke verschil tussen DEMO en OntoUML. In OntoUML staat het aanmaken van een instantie van de klasse 'Product' synoniem voor wat in DEMO het aanmaken van het Productiefeit behelst. Hetzelfde geldt voor het aanmaken van een instantie van de Coördinatie-actie klasse en wat in DEMO het Coördinatiefeit wordt genoemd. Daarom zijn beide concepten niet in de OntoUML-weergave van DEMO in Figuur 22 opgenomen, maar spelen conceptueel wel een essentiële rol in de betekenis van de DEMO-transactie.

6.5.5 Intentiesoorten

Uit de toelichting in paragraaf 6.5.4 blijkt dat het concept *Intentie* een belangrijke rol speelt in *Coördinatie-acties*. DEMO hanteert een vaste structuur van mogelijke *Intenties* die in Coördinatie-acties kunnen worden geuit, welke in Figuur 23 zijn weergegeven.

Elke *Transactie* bestaat uit een basispatroon van *Coördinatie-acties*:

- **Verzoek** is de *Intentie* die de *Initiator* uit naar de *Executor* om een *Product* te realiseren middels het uitvoeren van een *Transactie*.
- **Belofte** is de *Intentie* die de *Executor* uit naar de *Initiator* als deze wel instemt met het leveren van het *Product* dat verzocht is door de *Initiator*.
- **Afwijzing** is de *Intentie* die de *Executor* uit naar de *Initiator* als deze niet instemt met het leveren van het *Product* dat verzocht is door de *Initiator*.
- **Verklaring** is de *Intentie* die de *Executor* uit naar de *Initiator* als deze klaar is met het realiseren van het *Product* dat verzocht is door de *Initiator*.
- **Aanvaarding** is de *Intentie* die de *Initiator* uit naar de *Executor* als deze wel instemt met het *Product* dat de *Executor* heeft *Verklaard*.
- **Verwerping** is de *Intentie* die de *Initiator* uit naar de *Executor* als deze niet instemt met het *Product* dat de *Executor* heeft *Verklaard*.

Het complete patroon van een *Transactie* wordt gevormd door het basispatroon plus de *Coördinatie-acties* die deel uitmaken van het herroepingspatroon:

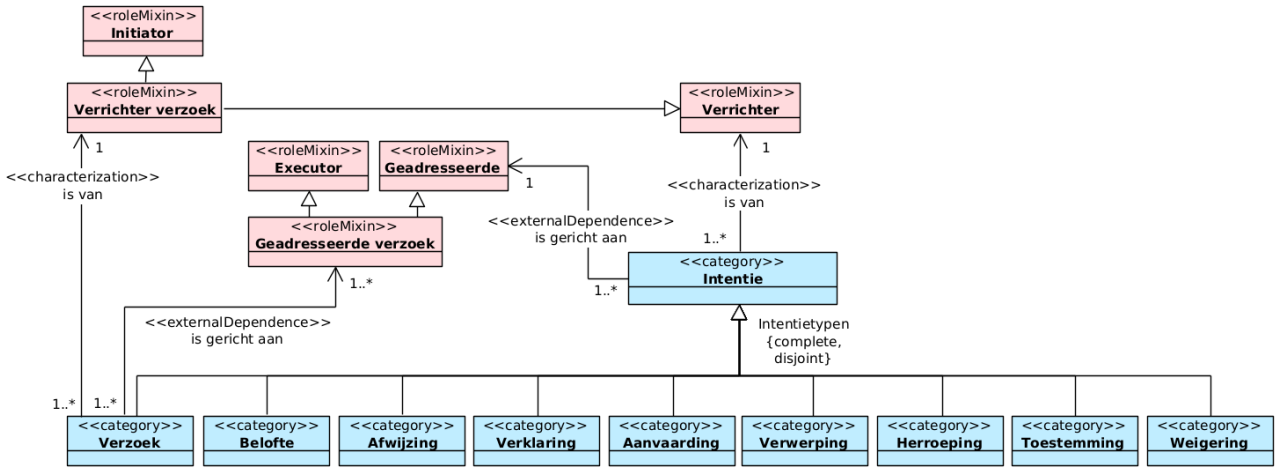
- **Herroeping** is de *Intentie* waarmee de *Verrichter* een eerder uitgevoerde Coördinatie-actie (*Verzoek*, *Belofte*, *Verklaring* en *Aanvaarding*) kan terugtrekken.
- **Toestemming** is de *Intentie* waarmee de *Geadresseerde* wel instemt met de *Herroeping* van de *Verrichter* van een *Coördinatie-actie*.
- **Weigering** is de *Intentie* waarmee de *Geadresseerde* niet instemt met de *Herroeping* van de *Verrichter* van een *Coördinatie-actie*.

Een voorbeeld van een *Herroeping* als onderdeel van het herroepingspatroon is:

- Initiator afronden verkoop (*Verrichter)* herroept (*Intentie)* Afronder verkoop (*Geadresseerde*) [Initiator afronden verkoop verzoekt Afronder verkoop Afronden verkoop is voltooid]

In een *Transactie* kunnen zowel de *Initiator* als de *Executor* optreden als zowel de *Verrichter* als de *Geadresseerde* van *Coördinatie-acties*. Daarbij zijn bepaalde *Intenties* exclusief voor de *Initiator* of *Executor*, terwijl andere *Intenties* door zowel de *Initiator* als de *Executor* kunnen worden gebruikt. Om duidelijk te maken dat een *Initiator* altijd de *Actor* is die als *Verrichter* een *Coördinatie-actie* met *Intentie* van *Verzoek* verstuurt richting de *Executor* als *Geadresseerde*, is in Figuur 23 is te zien dat *Verrichter verzoek* een specialisatie is van zowel *Initiator* als *Verrichter*, en dat *Geadresseerde verzoek* een specialisatie is van zowel *Executor* als *Geadresseerde*. Omwille van leesbaarheid is dit niet volledig uit gemodelleerd in Figuur 23 voor alle beperkingen op dit vlak:

- De *Intenties* die vanuit de *Initiator* als *Verrichter* middels een *Coördinatie-actie* richting de *Executor* als *Geadresseerde* verstuurd kunnen worden zijn: *Verzoek*, *Aanvaarding*, en *Verwerping*.
- De *Intenties* die vanuit de *Executor* als *Verrichter* middels een *Coördinatie-actie* richting de *Initiator* als *Geadresseerde* verstuurd kunnen worden zijn: *Belofte*, *Afwijzing*, *Uitvoering*, en *Verklaring*.
- De Intenties die zowel door *Initiator* als *Executor* middels een Coördinatie-actie geuit kunnen worden zijn: *Herroeping*, *Toestemming*, en *Weigering*.

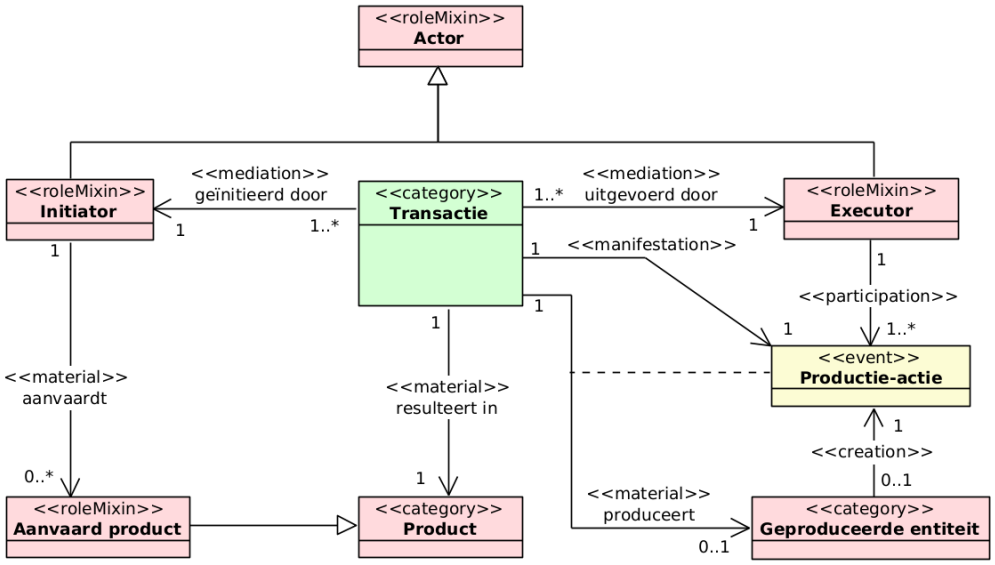


Figuur 23: Mogelijke intentiesoorten die in coördinatie-acties kunnen worden geuit door de verrichter.

6.5.6 Het metamodel van een transactie (Productie-perspectief)

Gedurende de *Executiefase* van een *Transactie* vinden *Productie-acties* die de *Executor* uitvoert plaats om de *Geproduceerde entiteit* te realiseren. Ten opzichte van Figuur 20 en Figuur 22 voegt Figuur 24 een aantal nieuwe begrippen toe die te maken hebben met de totstandkoming van de *Geproduceerde entiteit* door de *Executor*:

- **Geproduceerde entiteit** is een entiteit dat het concrete resultaat is van de *Productie-actie* van een *Transactie*. Een voorbeeld hiervan is: `Verkoop XY74921`.
- **Productie-actie** is de actie in een *Transactie* waarin de *Executor* de *Geproduceerde entiteit* van de *Transactie* tot stand brengt ofwel creëert. Hiermee is de *Executiefase* van de *Transactie* afgerond. Het daadwerkelijke (eind)*Product* van een *Transactie* begint pas te bestaan zodra de *Initiator* de *Aanvaarding* heeft verricht en de *Resultaatfase* is afgerond.



Figuur 24: Ontologie van een transactie vanuit het Productie-perspectief

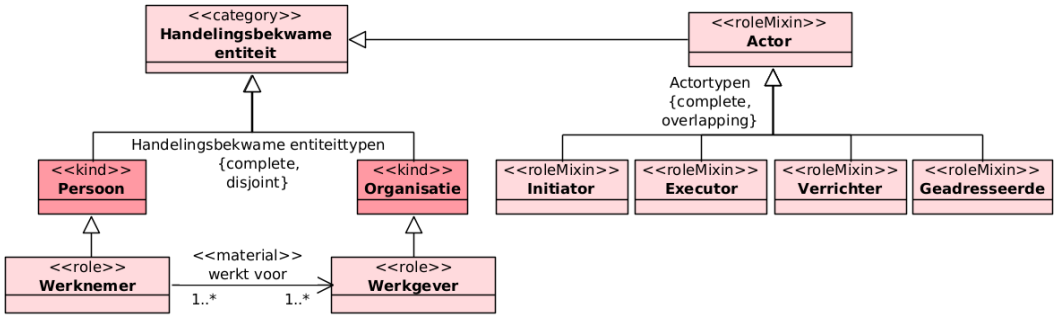
6.5.7 Actoren

Zoals Figuur 25 weergeeft, zijn *Initiator, Executor, Verrichter* en *Geadresseerde* een specialisatie van de «roleMixin» klasse *Actor*. In paragraaf 6.3.2 is Actor gedefinieerd als: een persoon of groep van personen die een rol vervult in een *Transactie*. Zoals Figuur 25 ook weergeeft zijn zowel `Persoon` als `Organisatie` vormen van een `Handelingsbekwame entiteit` die een mentale houding heeft en in staat is om acties uit te voeren

en gebeurtenissen waar te nemen[1]. De beperking 'Handelingsbekwame entiteittypen' in Figuur 25 specificeert dat dit de enige twee typen zijn die worden gehanteerd (aangegeven door 'complete'), waarbij een instantie altijd van één van de twee typen is (aangegeven door 'disjoint').

In dit domeinmodel is iedere instantie van de `Werknemer` «role» klasse een specialisatie van de «kind» klasse `Persoon` en iedere instantie van de `Werkgever` «role» klasse een specialisatie van de «kind» klasse `Organisatie`, waarbij een `Werknemer` altijd werkt voor minimaal één `Werkgever`. Een privaat persoon (bijv. een mens waarvoor een laboratoriumonderzoek moet worden uitgevoerd) is een instantie van de «kind» klasse `Persoon`.

Een `Handelingsbekwame entiteit` kan verschillende rollen vervullen binnen het domein, namelijk *Initiator* en *Executor* van *Transacties*, maar ook *Verrichter* en *Geadresseerde* van *Coördinatie-acties*.



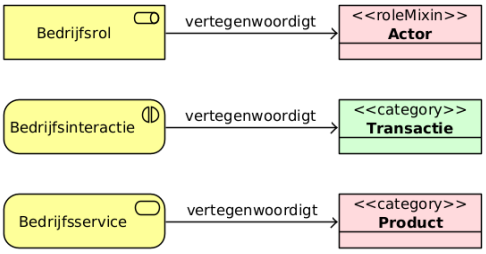
Figuur 25: Persoon en Organisatie in relatie tot Actor

6.6 Mapping van beschrijvend naar formeel

Deze paragraaf vormt een brugfunctie van de informele, beschrijvende weergave van het sub-domein Diagnostiek, naar de formele specificatie in paragraaf 6.7.

Figuur 26 geeft de mapping van de beschrijvende elementen van ArchiMate naar de formele constructen uit DEMO, weergegeven in OntoUML:

- De *Actor* klasse met «roleMixin» stereotype is de formele vertegenwoordiging van de informele *Bedrijfsrol*.
- De *Transactie* klasse met «category» stereotype is de formele vertegenwoordiging van de informele *Bedrijfsinteractie*. De *Transactie* «category» kan alleen worden gespecialiseerd door klassen met het stereotype «relaton».
- De *Product* klasse met «kind» stereotype is de formele vertegenwoordiging van de informele *Bedrijfsservice*.

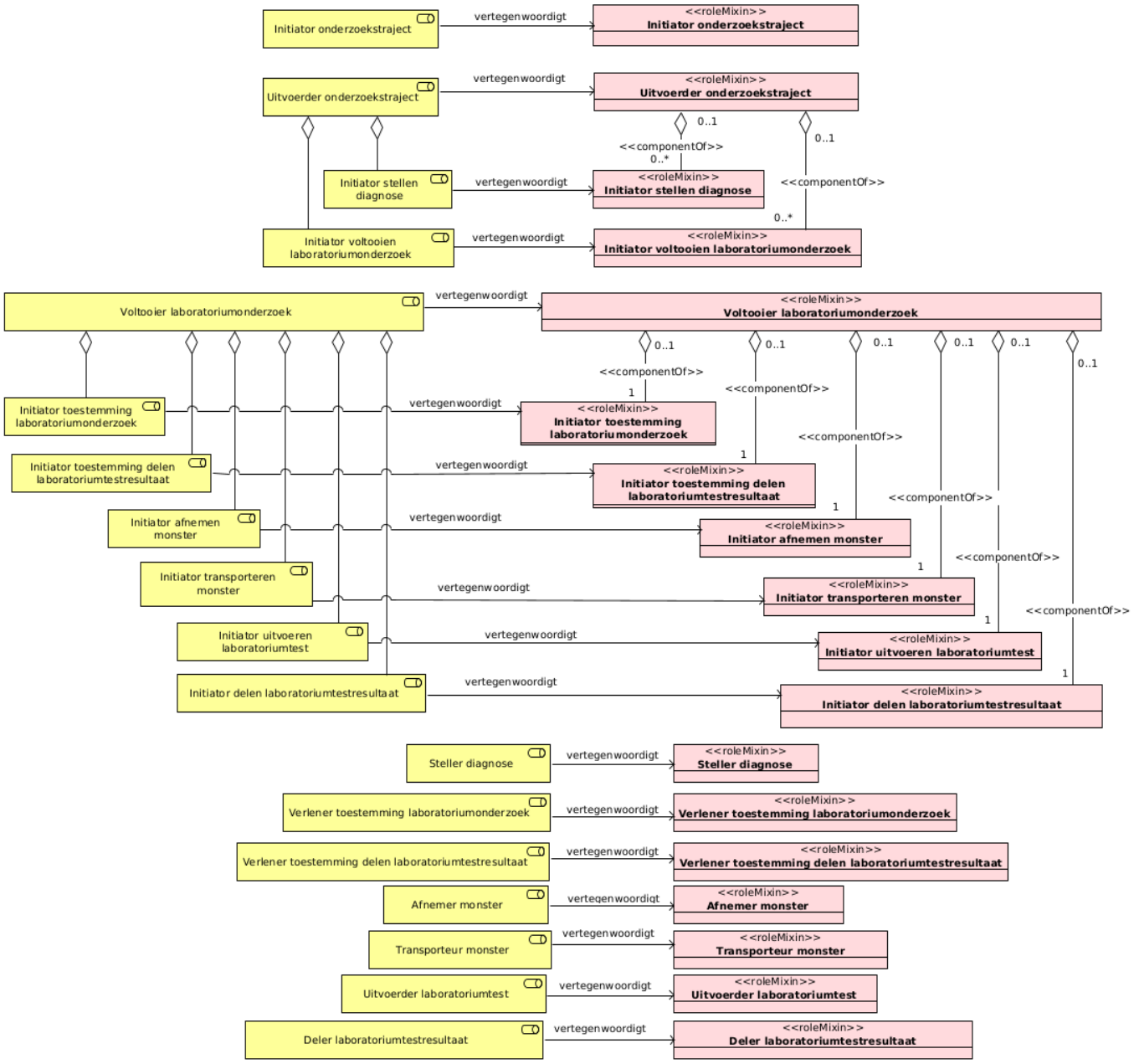


Figuur 26: Mapping van ArchiMate naar formele DEMO constructen weergegeven in OntoUML stereotypen

[1] Een Software/AI agent kan ook als Handelingsbekwame entiteit optreden, waarbij deze nooit de eindverantwoordelijkheid kan dragen voor de handelingen die het verricht.

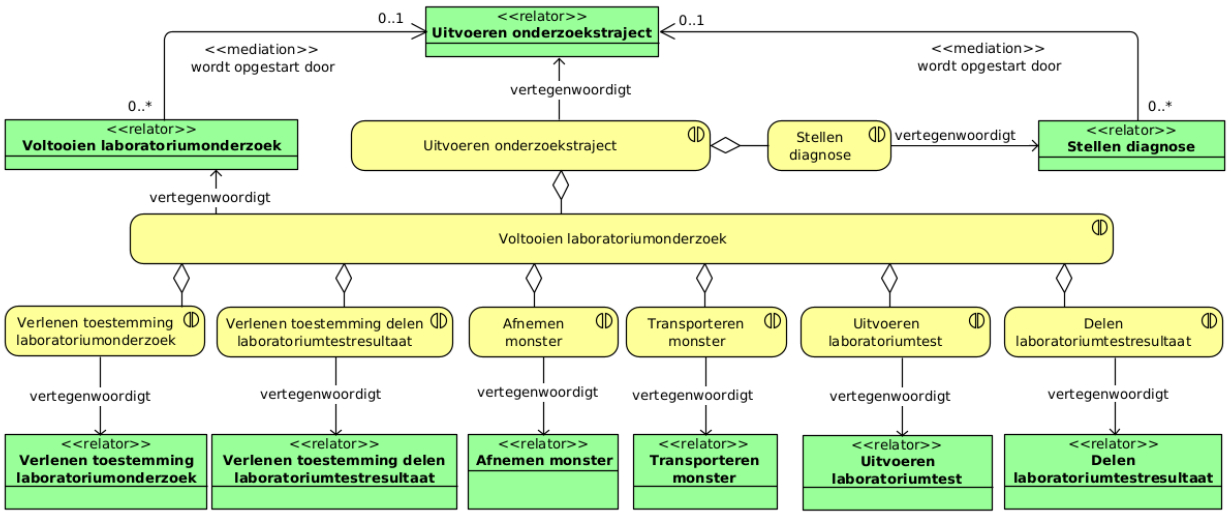
In de rest van paragraaf 6.6 wordt een mapping gegeven van de beschrijvingen uit paragraaf 6.4 naar de formele specificaties in paragraaf 6.7.

6.6.1 Van Bedrijfsrollen naar Actoren



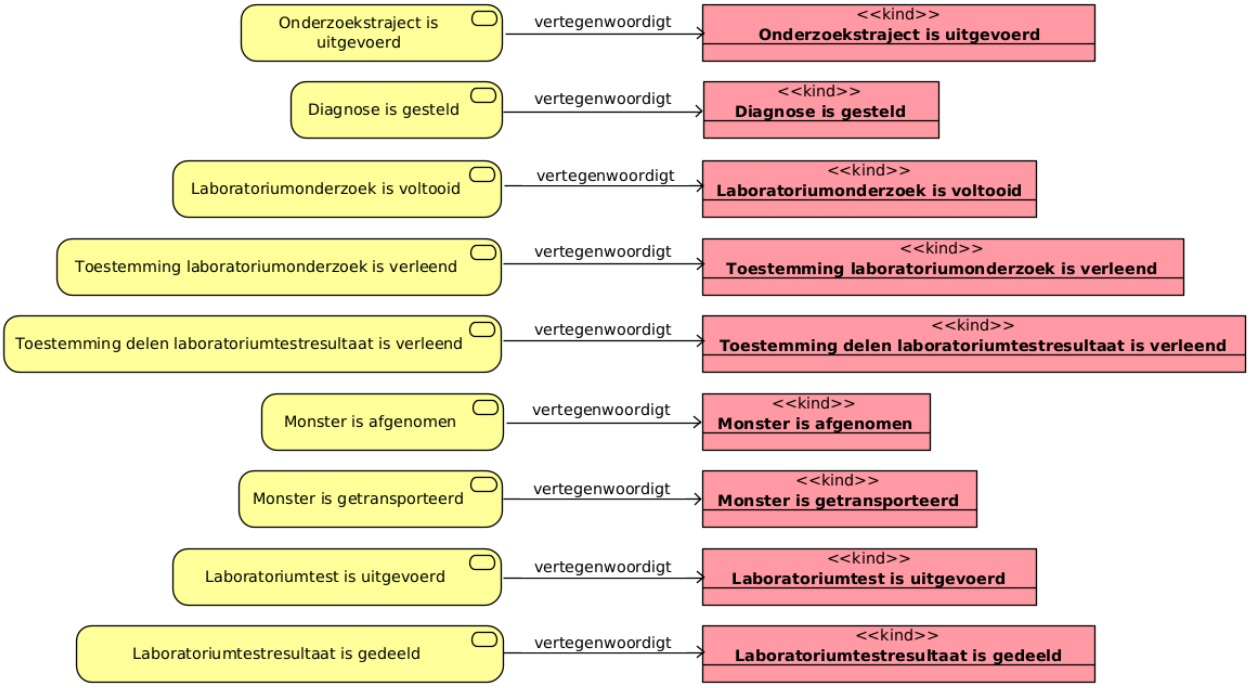
Figuur 27: Mapping van Bedrijfsrollen naar Actoren

6.6.2 Van Bedrijfsinteracties naar Transacties



Figuur 28: Mapping van Bedrijfsinteracties naar Transacties

6.6.3 Van Bedrijfsservices naar Producten

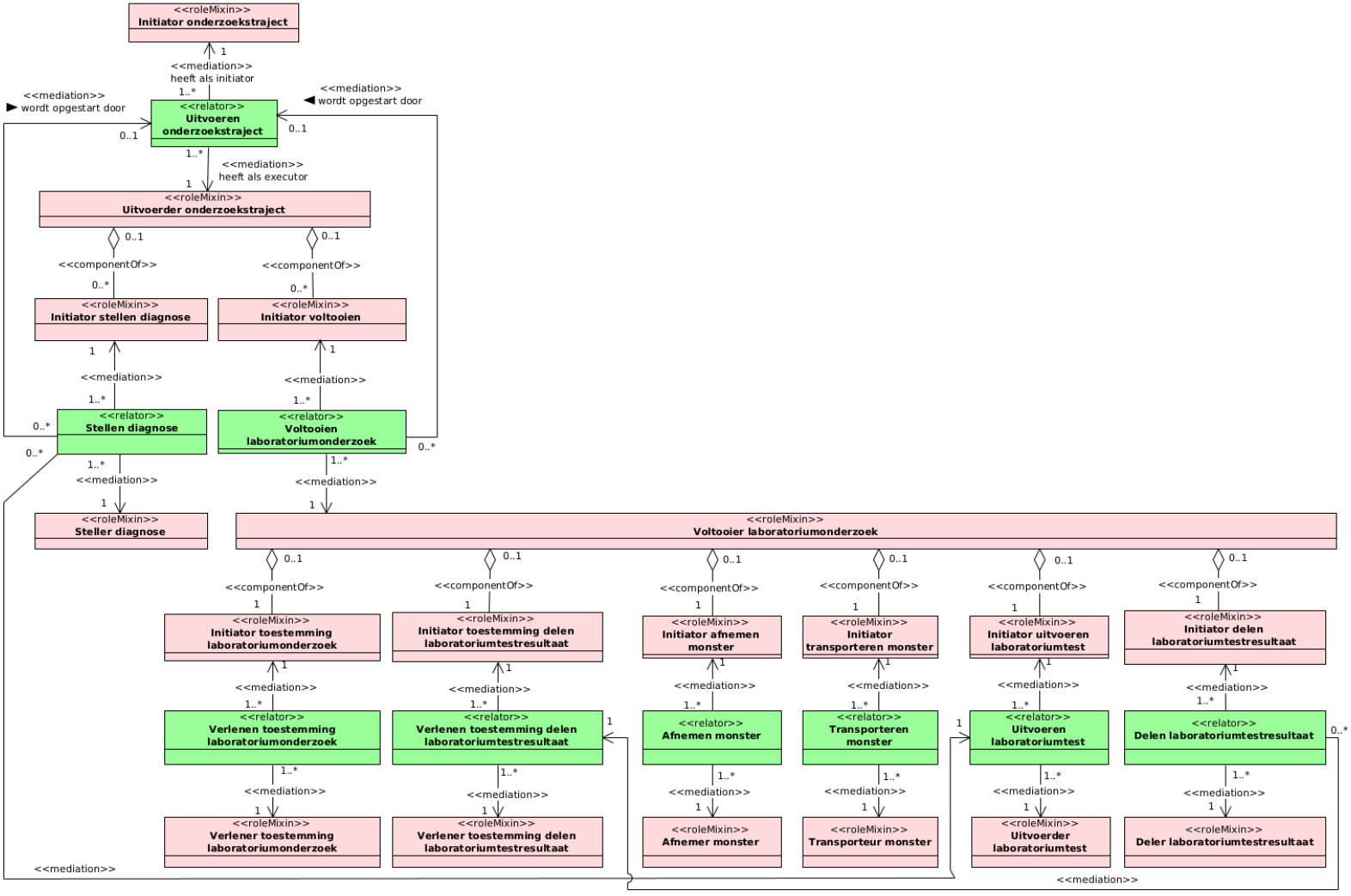


Figuur 29: Mapping van Bedrijfsservices naar Producten

6.7 Formele specificaties

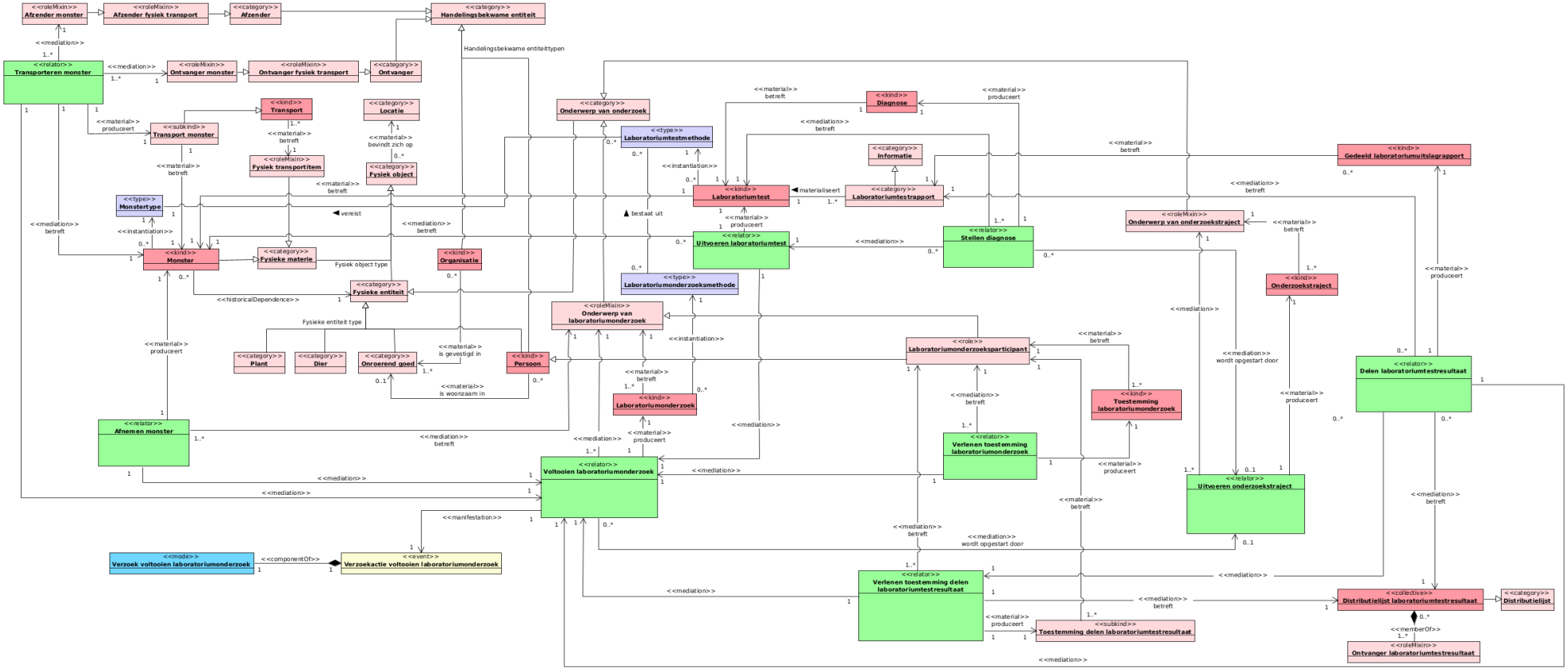
6.7.1 Overzichten

6.7.1.1 Coördinatiestructuur Diagram



Figuur 30: Volledig Coördinatiestructuur Diagram

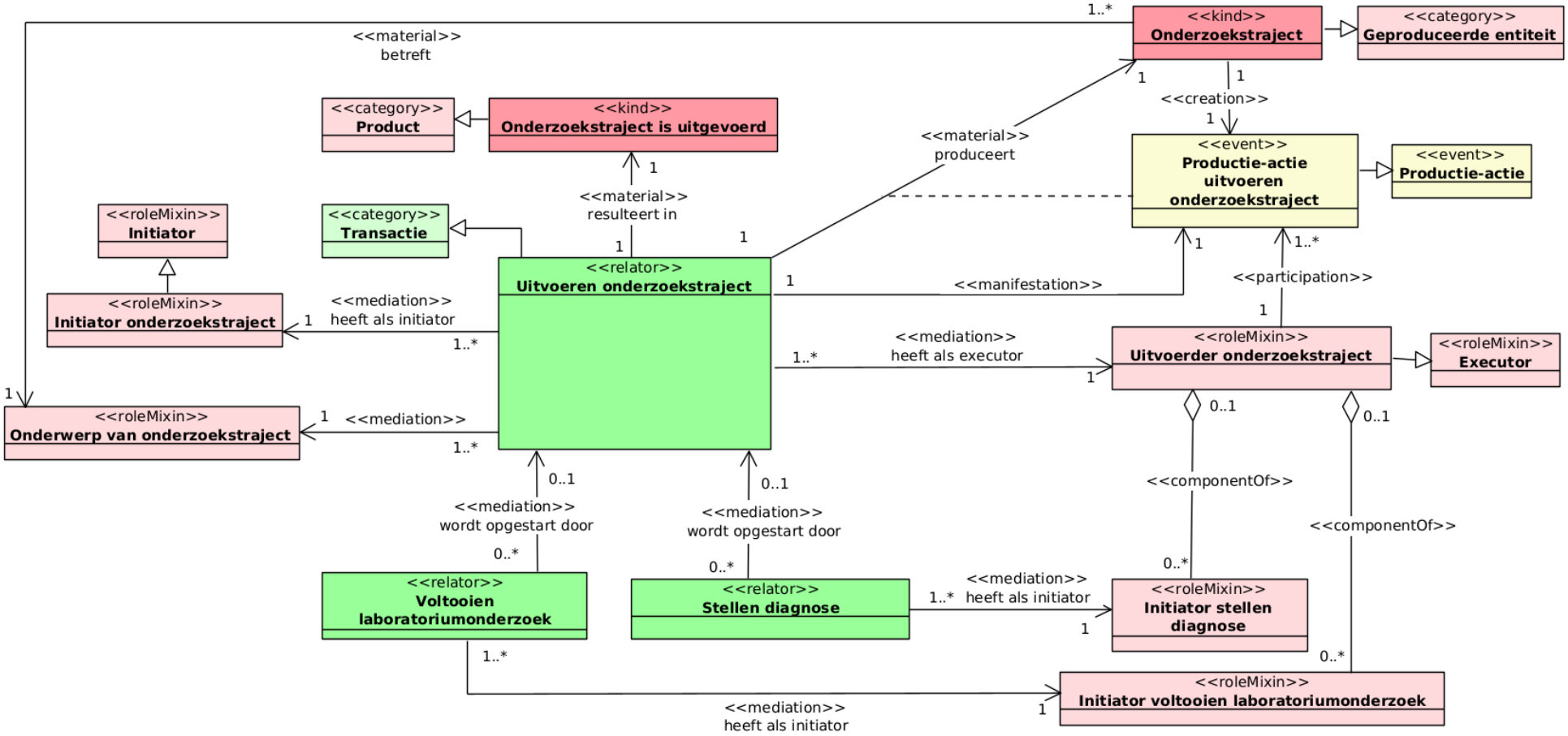
6.7.1.2 Informatiemodel



Figuur 31: Volledig informatiemodel

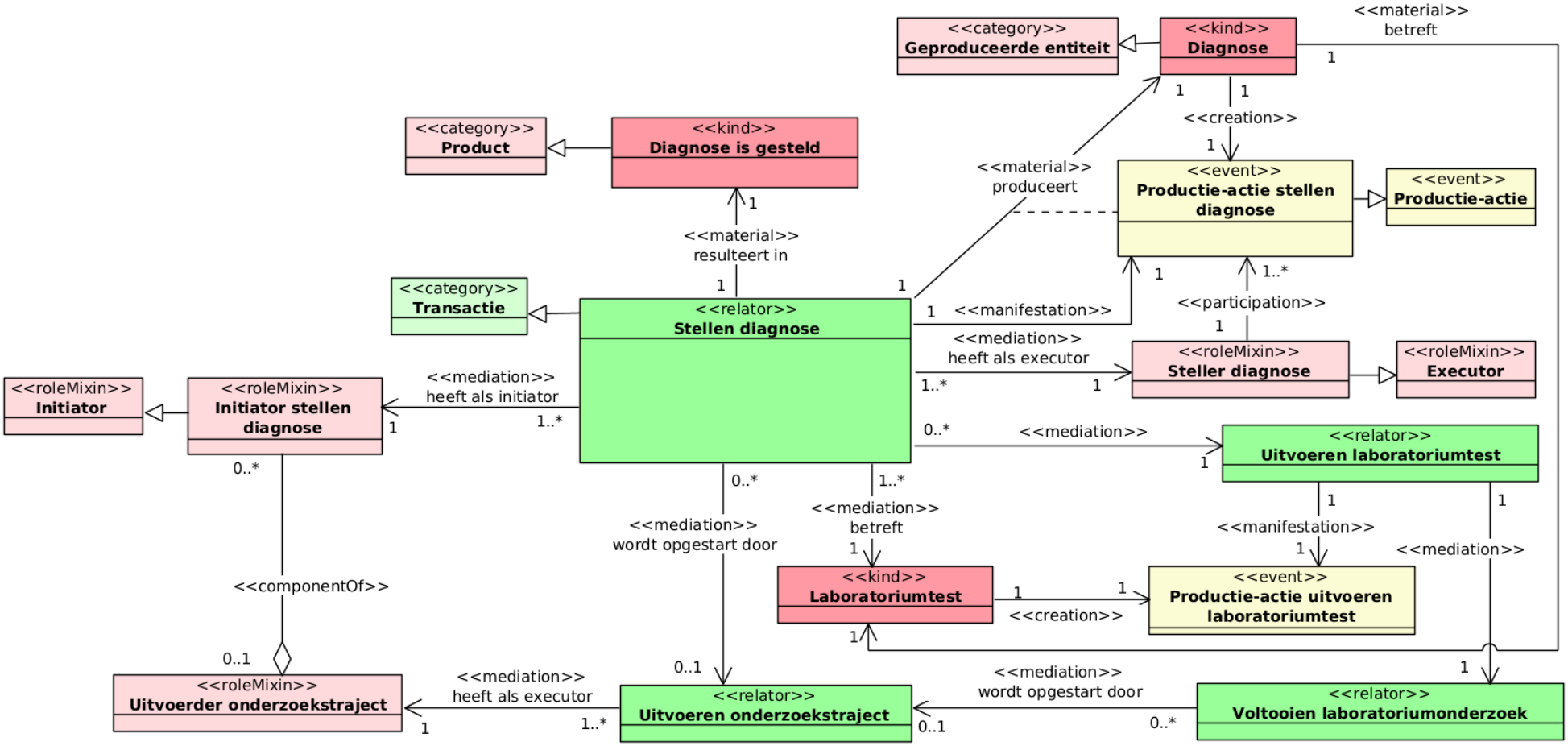
6.7.2 Detailmodellen per transactie

6.7.2.1 Uitvoeren onderzoekstraject



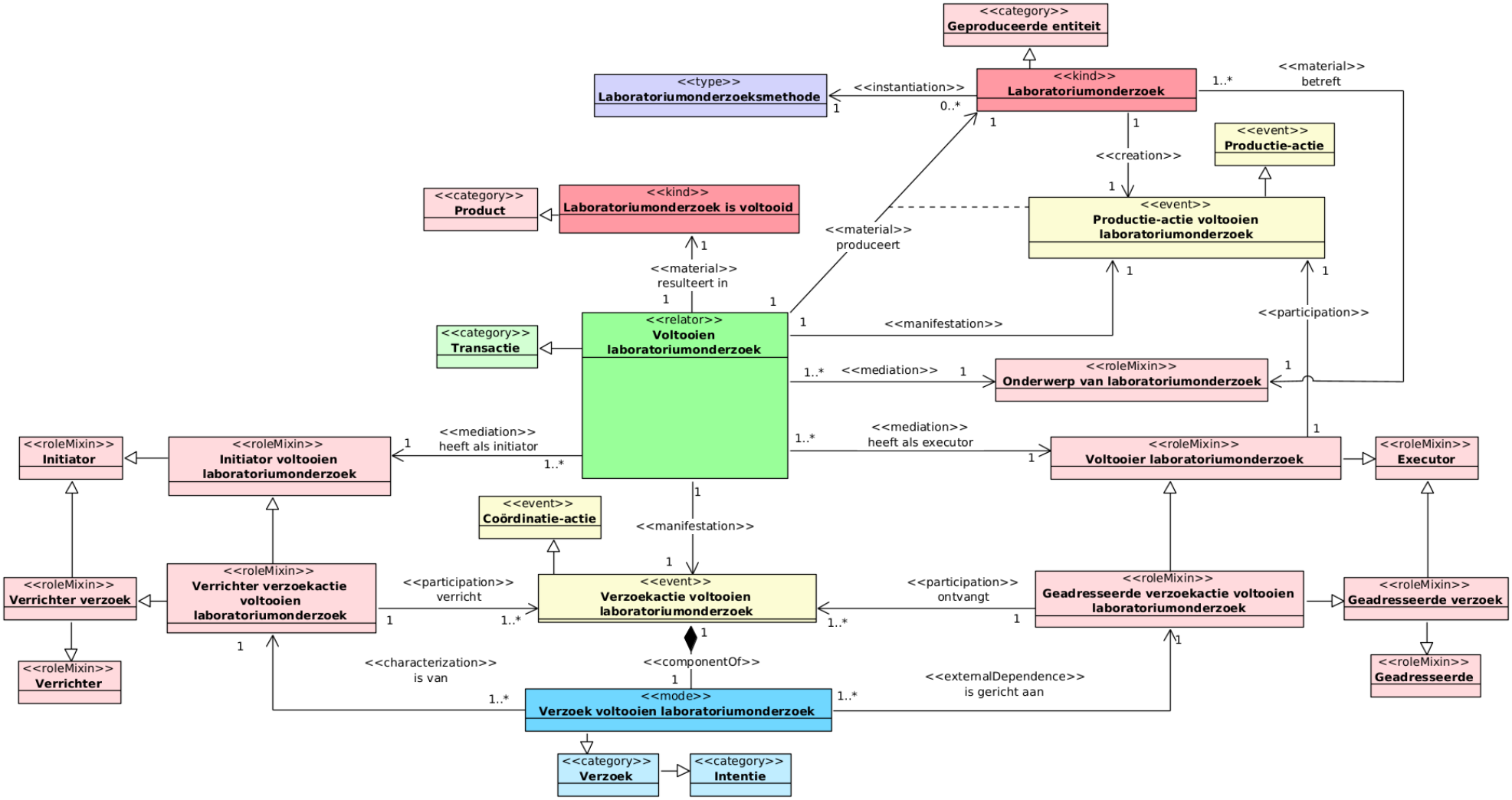
Figuur 32: Details van Uitvoeren onderzoekstraject

6.7.2.2 *Stellen diagnose*



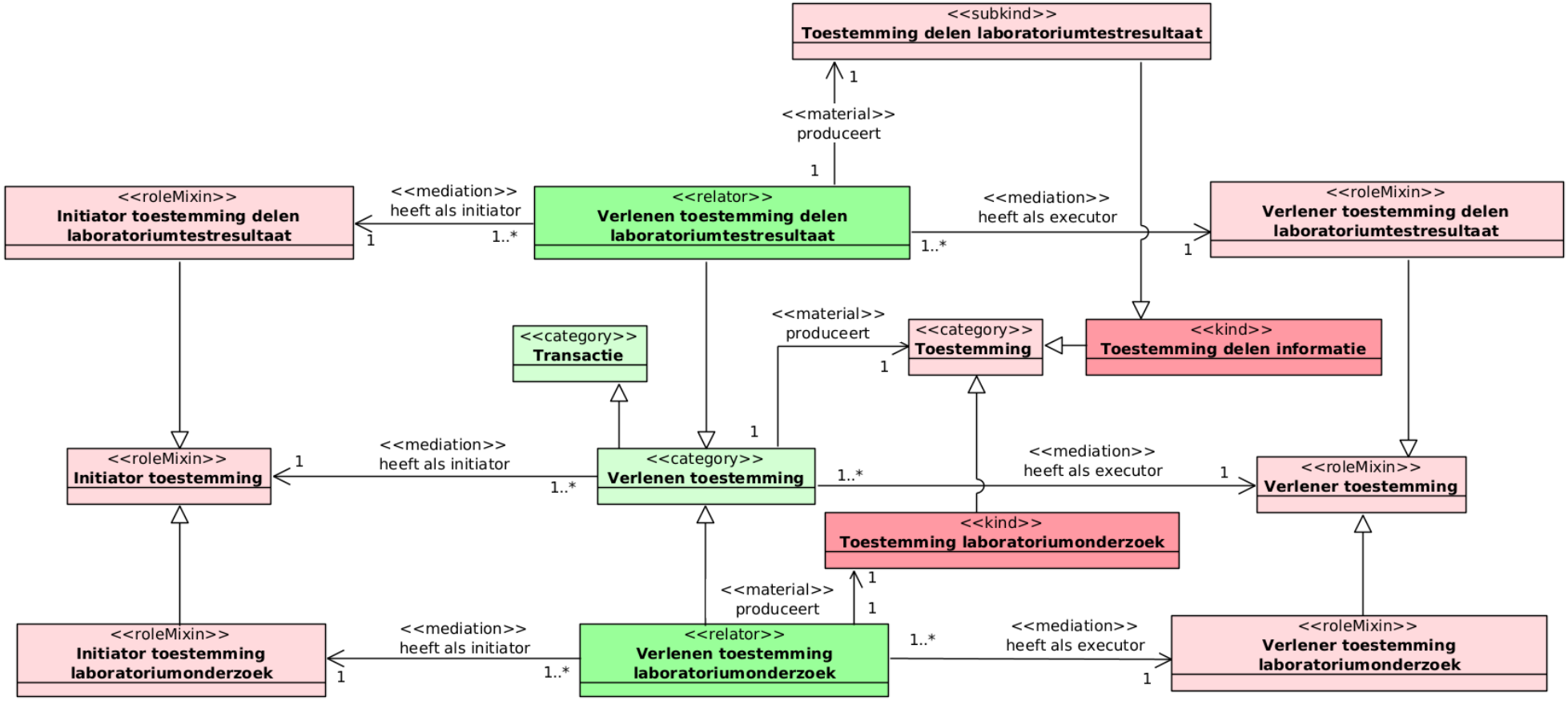
Figuur 33: Details Stellen diagnose

6.7.2.3 *Voltooien laboratoriumonderzoek*



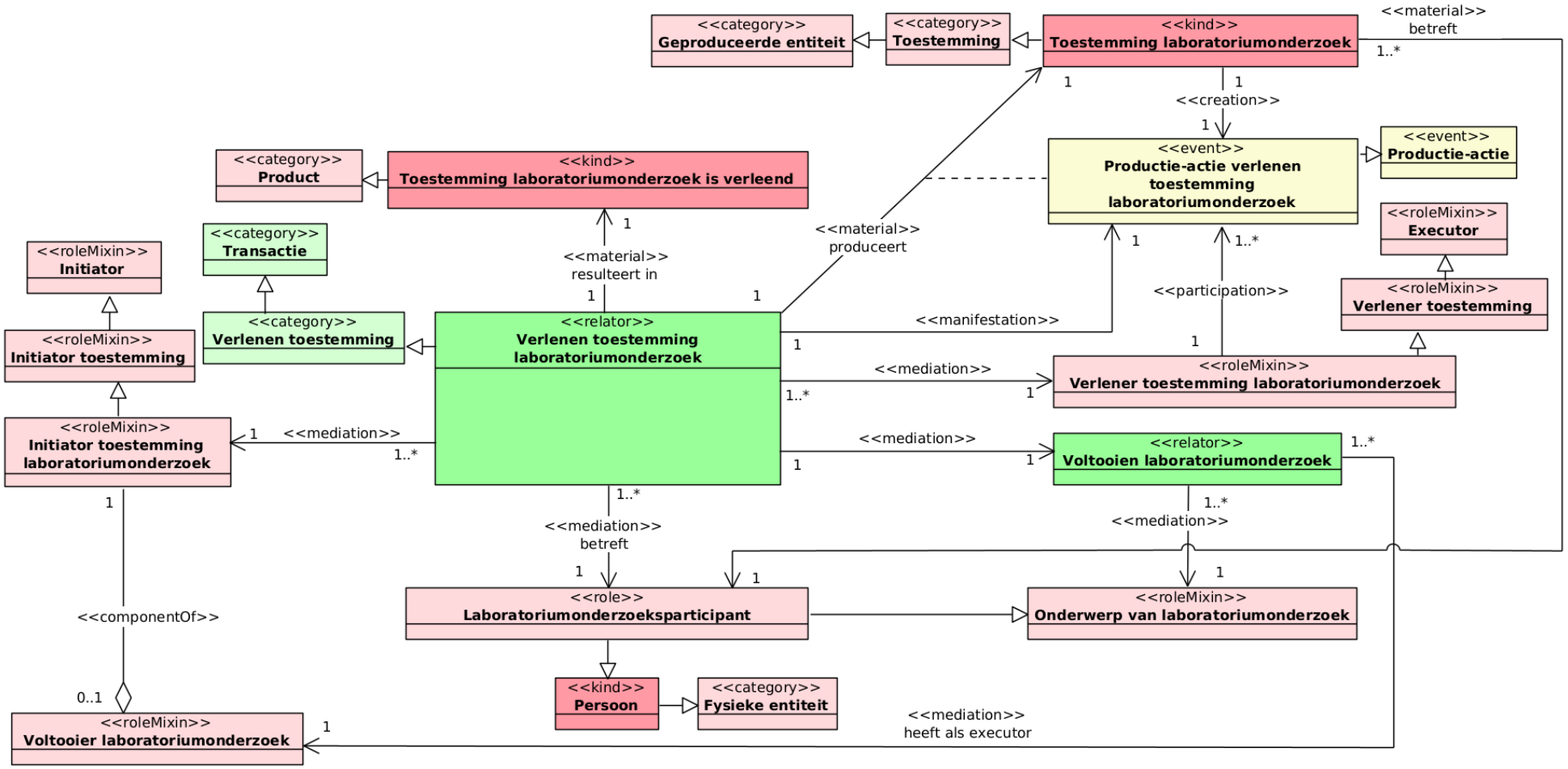
Figuur 34: Details Voltooien laboratoriumonderzoek

6.7.2.4 *Verlenen toestemming*



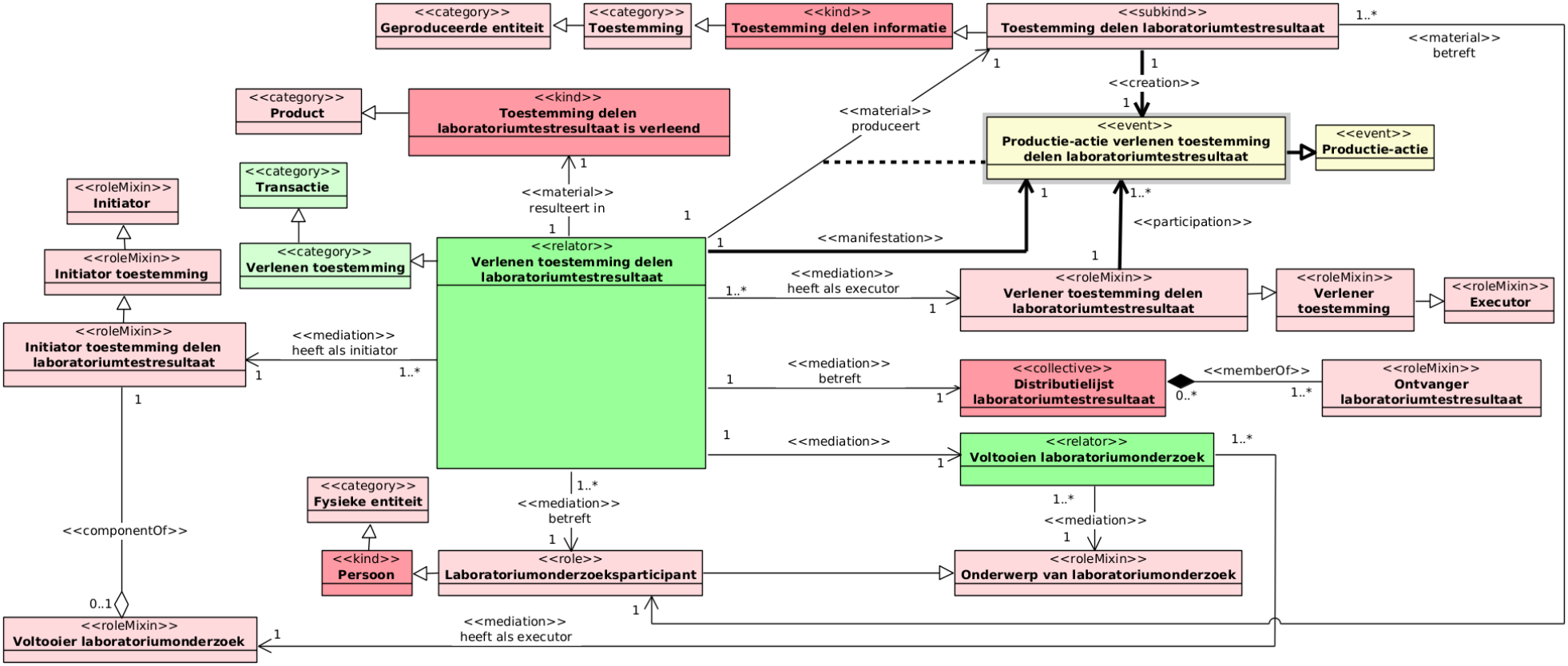
Figuur 35: Details basispatroon Verlenen toestemming

6.7.2.5 *Verlenen toestemming laboratoriumonderzoek*



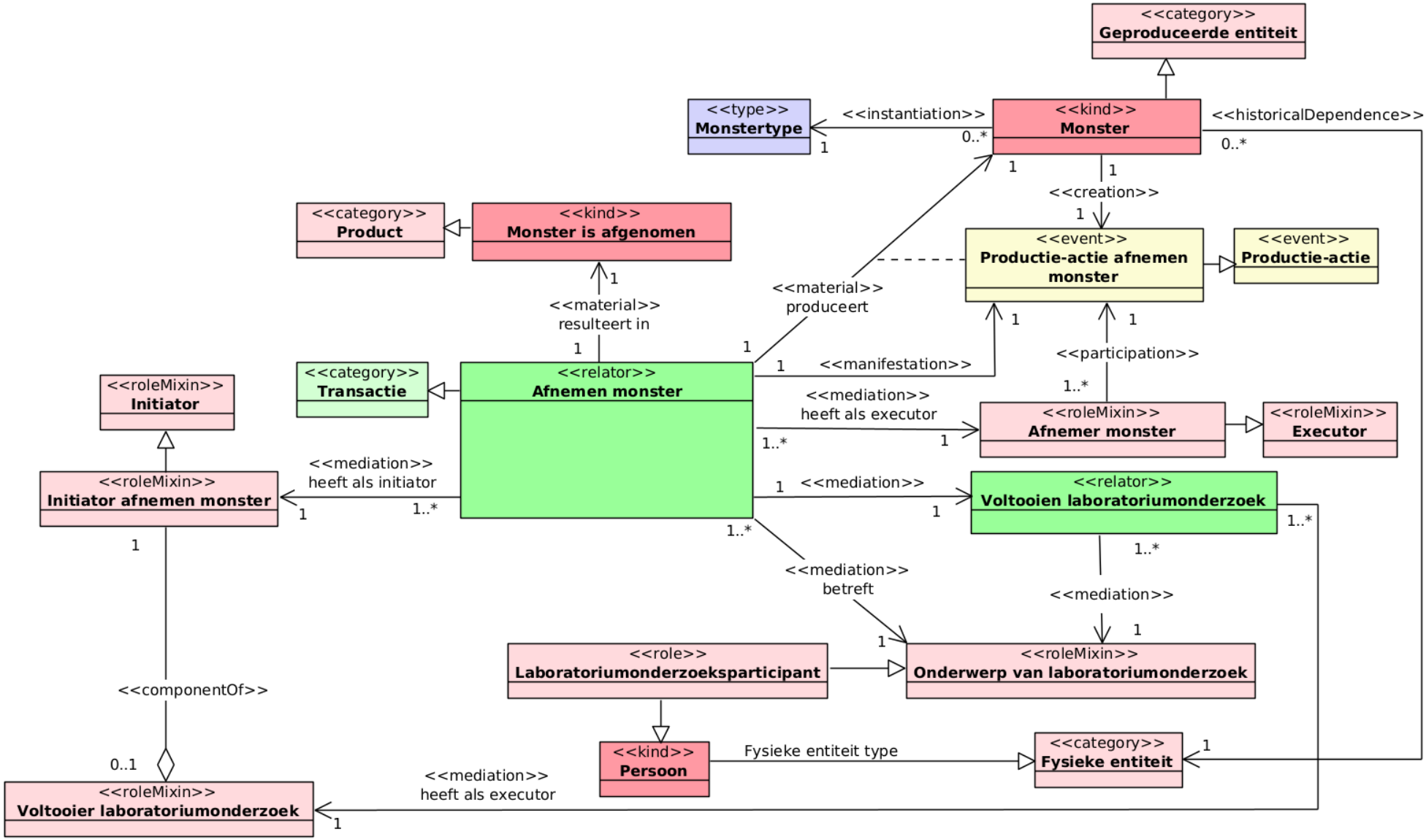
Figuur 36: Details Verlenen toestemming laboratoriumonderzoek

6.7.2.6 *Verlenen toestemming delen laboratoriumtestresultaat*



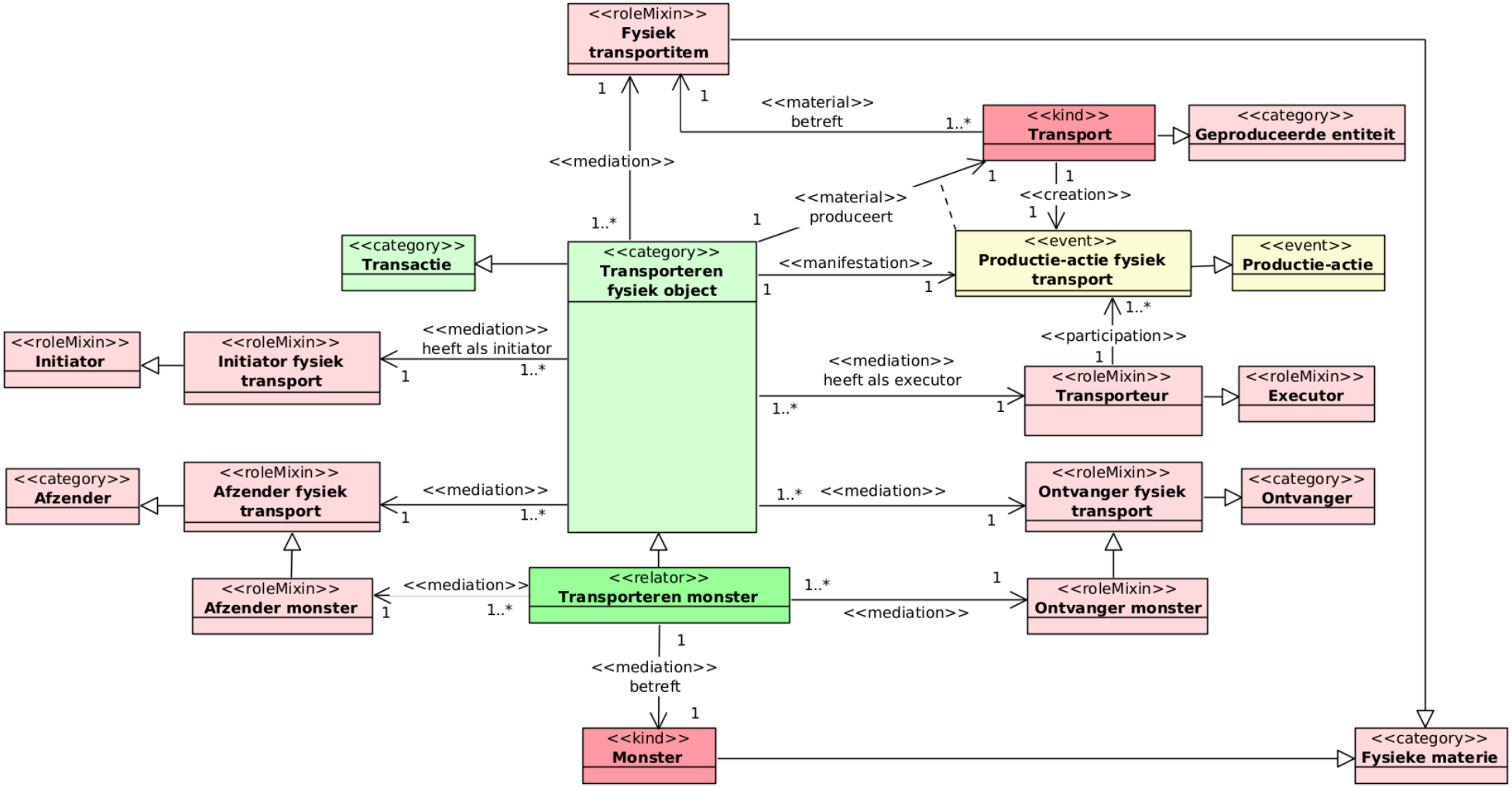
Figuur 37: Details Verlenen toestemming delen laboratoriumtestresultaat

6.7.2.7 Afnemen monster



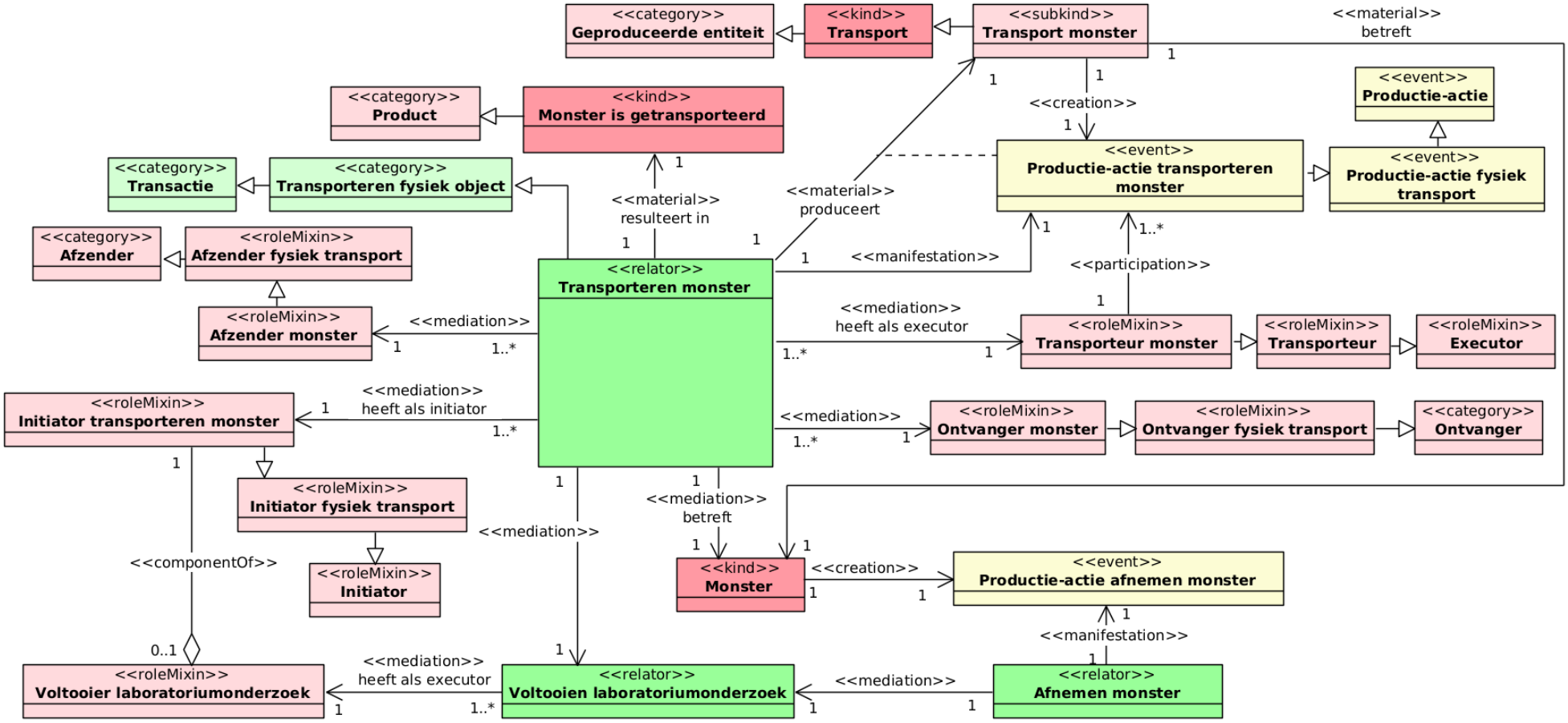
Figuur 38: Details Afnemen monster

6.7.2.8 *Transporteren fysiek object*



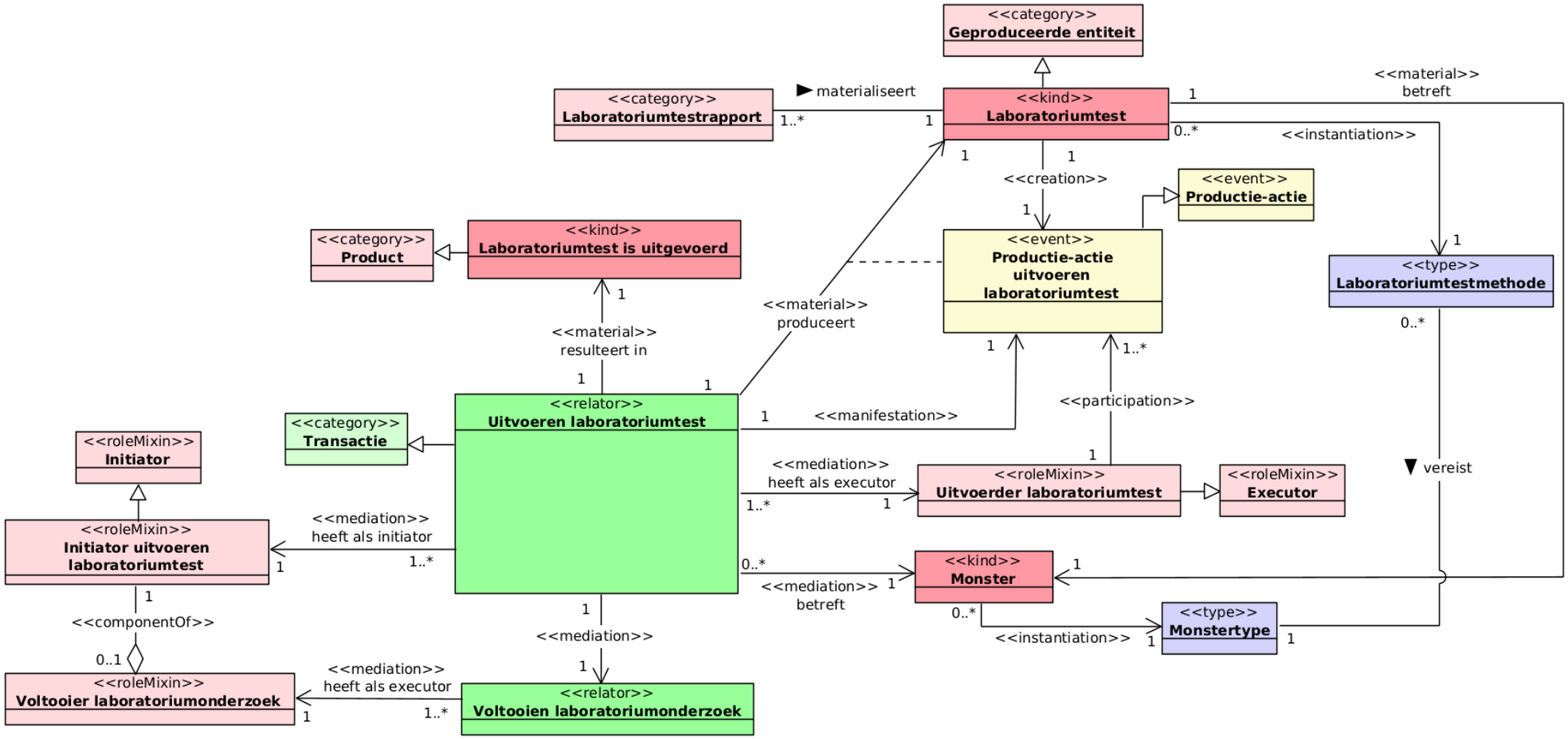
Figuur 39: Details basispatroon Transporteren fysiek object

6.7.2.9 *Transporteren monster*



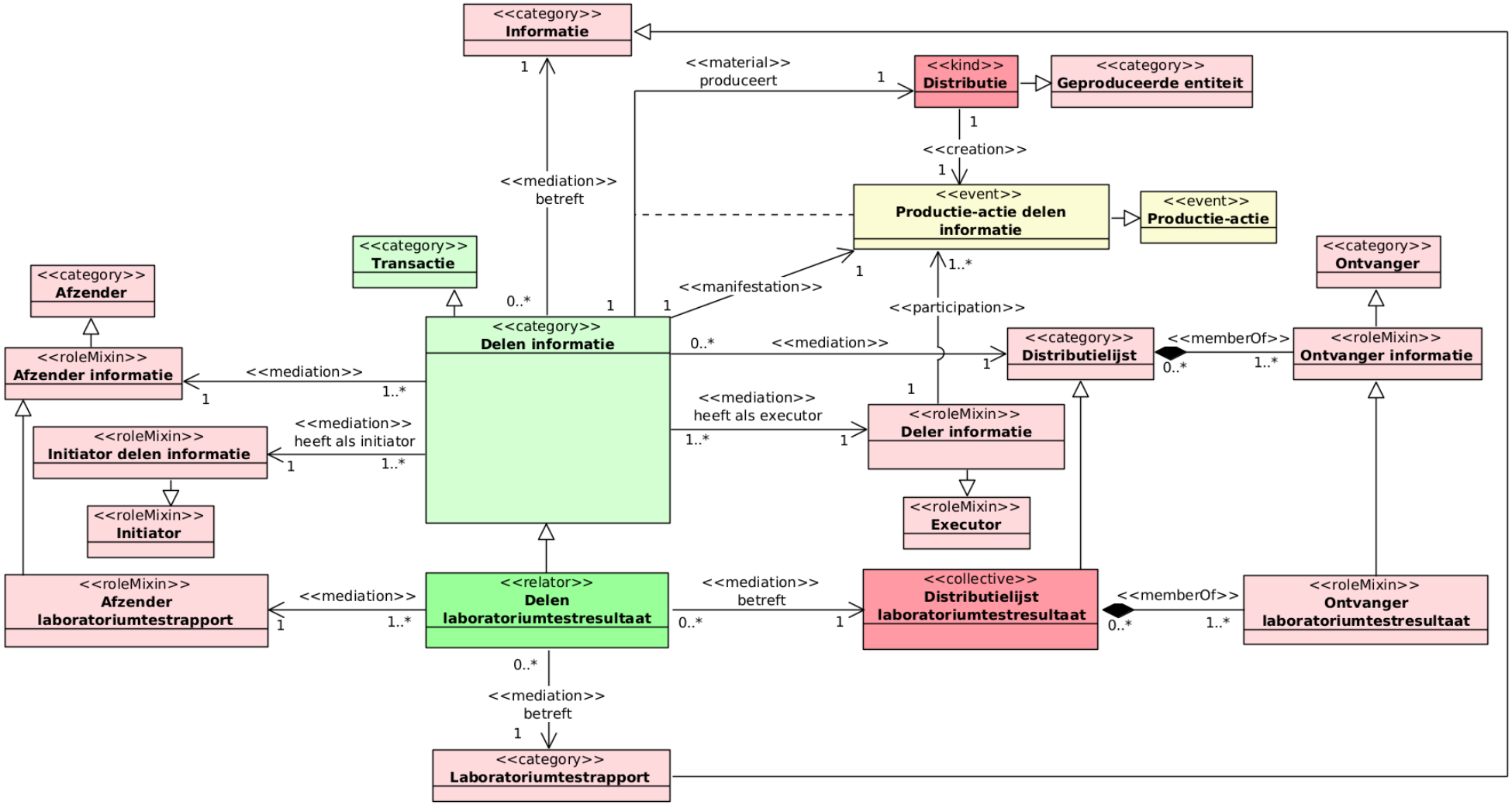
Figuur 40: Details Transporteren monster

6.7.2.10 Uitvoeren laboratoriumtest



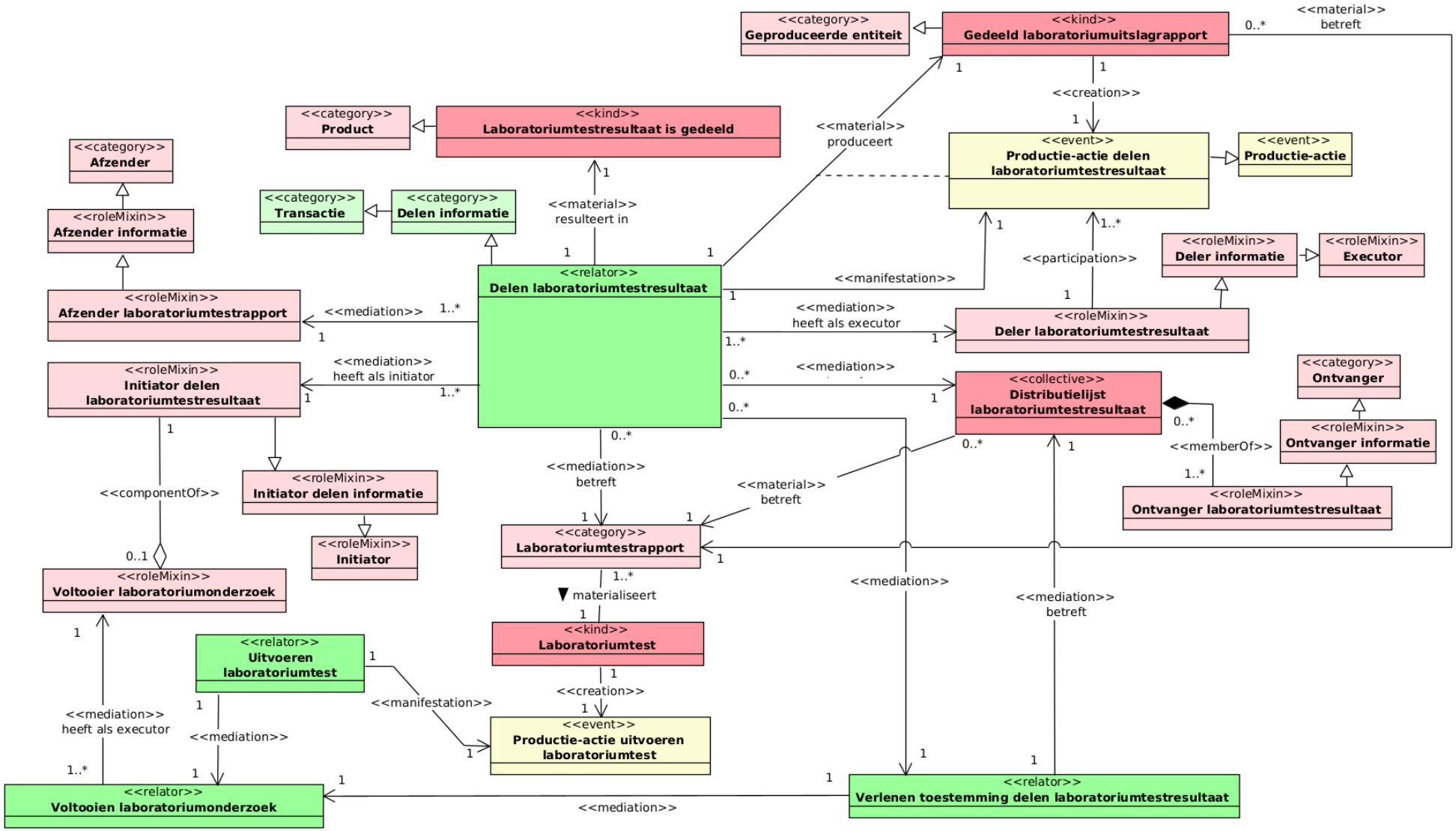
Figuur 41: Details Uitvoeren laboratoriumtest

6.7.2.11 Delen informatie



Figuur 42: Details basispatroon Delen informatie

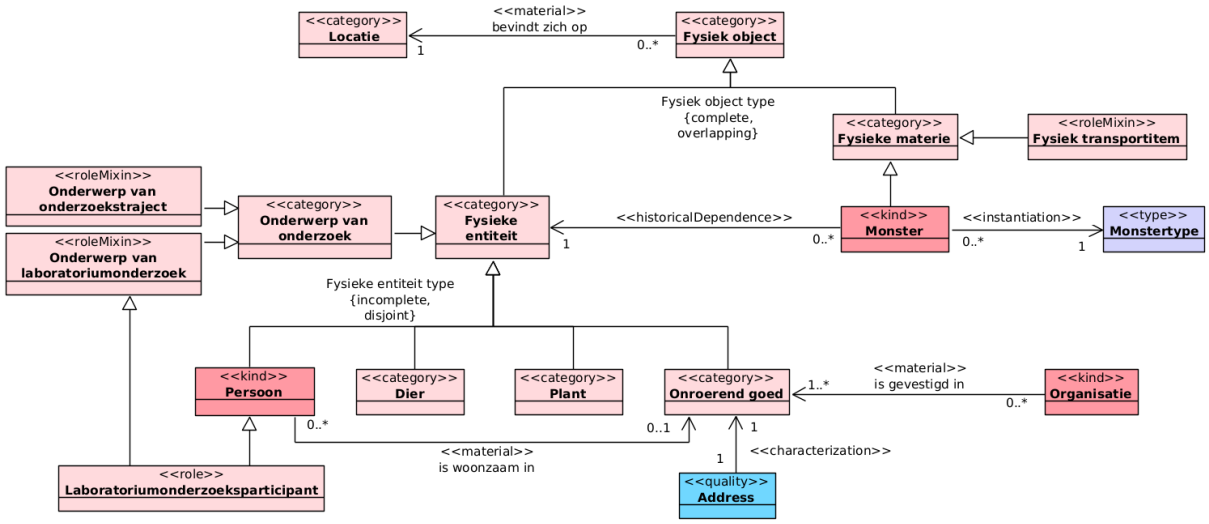
6.7.2.12 Delen laboratoriumtestresultaat



Figuur 43: Delen laboratoriumtestresultaat

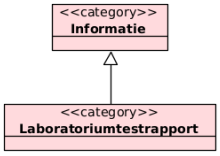
6.7.3 Rigide sortals

6.7.3.1 Fysieke objecten



Figuur 44: Fysieke objecten

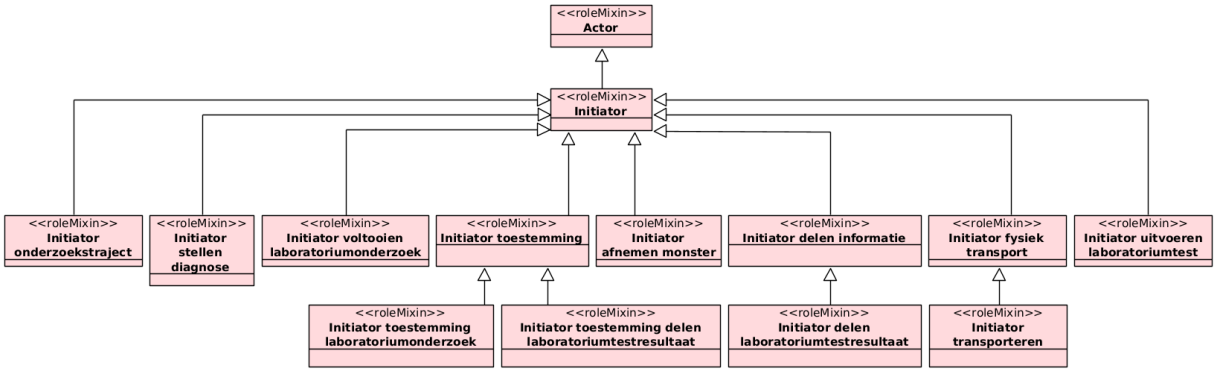
6.7.3.2 Informatie



Figuur 45: Informatie

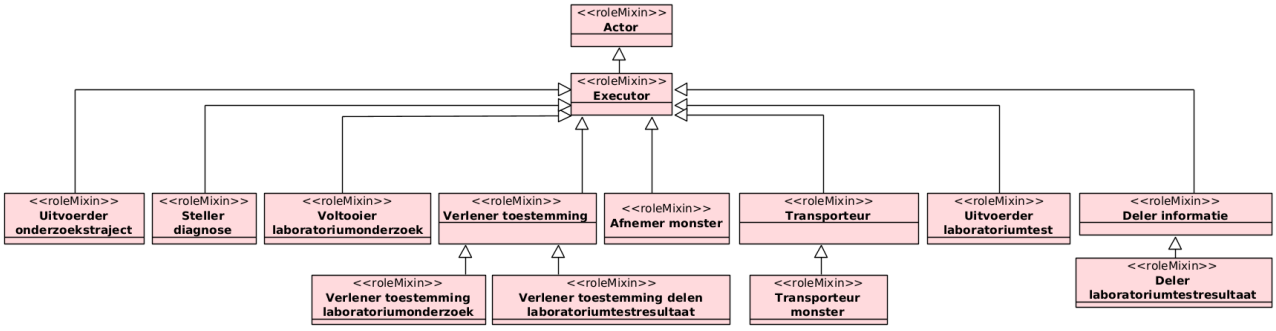
6.7.4 Roles & RoleMixins

6.7.4.1 Initiators



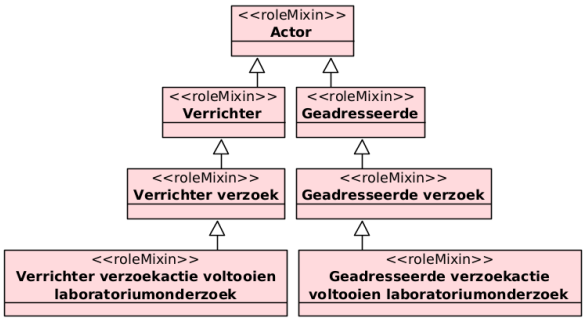
Figuur 46: Initiators weergegeven als roleMixins

6.7.4.2 Executors



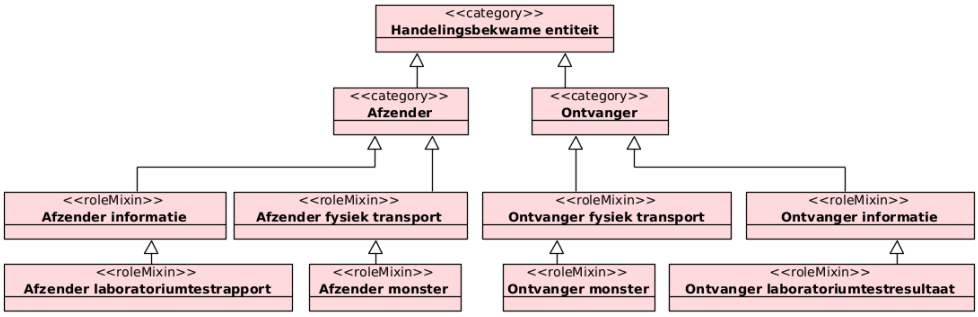
Figuur 47: Executors weergegeven als roleMixins

6.7.4.3 *Verrichters en Geadresseerden*



Figuur 48: Verrichters en Geadresseerden weergegeven als roleMixins

6.7.4.4 *Afzenders en Ontvangers*



Figuur 49: Afzenders en Ontvangers weergegeven als roleMixins

7 Mapping formele specificaties met openEHR

In dit hoofdstuk wordt de vertaling gemaakt van de formele specificaties zoals beschreven in hoofdstuk 6 naar openEHR, de datastandaard die wordt gebruikt voor het opslaan en beheren van IZB-gerelateerde gegevens. In paragraaf 7.1 wordt een overzicht van de relevante archetypes uit openEHR weergegeven. In paragraaf 7.2 wordt per archetype een mapping gegeven naar de relevante klassen uit de formele specificaties uit hoofdstuk 6.

7.1 Relevante openEHR archetypes

In deze paragraaf wordt een opsomming gegeven van de archetypes van de openEHR standaard die relevant zijn in het Diagnostiek domein. Hierin wordt alleen een korte definitie (in het Engels) gegeven, zoals gehanteerd binnen de openEHR standaard. Tevens wordt een link gegeven naar de gedetailleerde informatie van het betreffende archetype die is gepubliceerd in de Clinical Knowledge Manager van openEHR: https://ckm.openehr.org/ckm/

Alvorens de individuele archetypes worden vermeld, geeft Figuur 50 een overzicht van alle relevante archetypes en de onderliggende datastructuur, weergegeven als UML-klassendiagram.

7.1.1 Laboratory test result

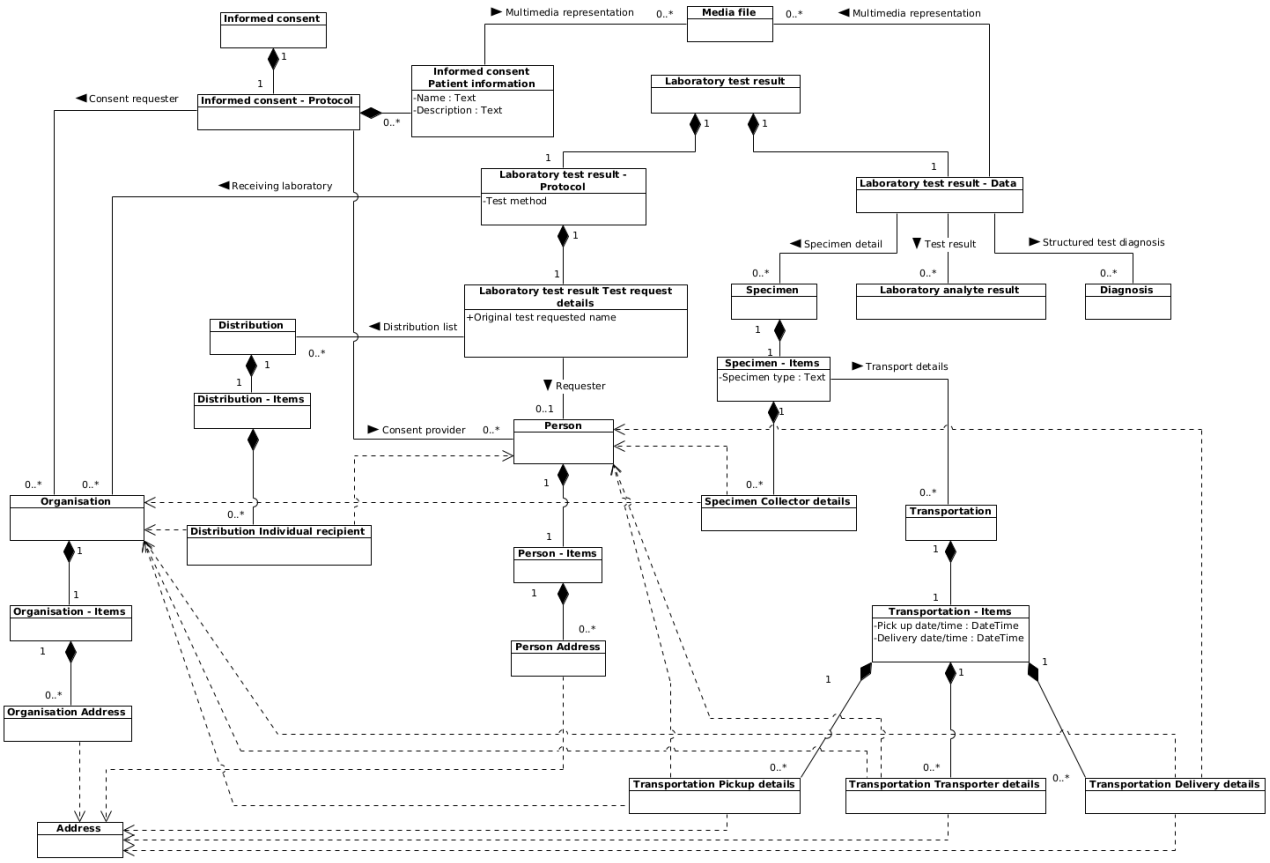
Definitie: "*The result, including findings and the laboratory's interpretation, of an investigation performed on specimens collected from an individual or related to that individual.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.2191

7.1.2 Distribution

Definitie: "*Details of the target of communication distribution, whether to identified individual parties or as a category.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.1591



Figuur 50: Datastructuur die de samenhang van de relevante openEHR archetypes weergeeft in UML

7.1.3 Informed consent

Definitie: "*Record of status and details of informed consent from an individual (or the individual's agent/proxy) for a proposed procedure, trial or other healthcare-related activity (including treatments and investigations), based upon a clear appreciation and understanding of the facts, implications, and possible future consequences by the consenting party.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.1303

7.1.4 Organisation

Definitie: "*An entity comprising one or more people and having a particular purpose.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.371

7.1.5 Person

Definitie: "*An individual human being.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.5358

7.1.6 Specimen

Definitie: "*A physical sample collected from, or related to, an individual for the purpose of investigation, examination or analysis.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.331

7.1.7 Transportation (of an item)

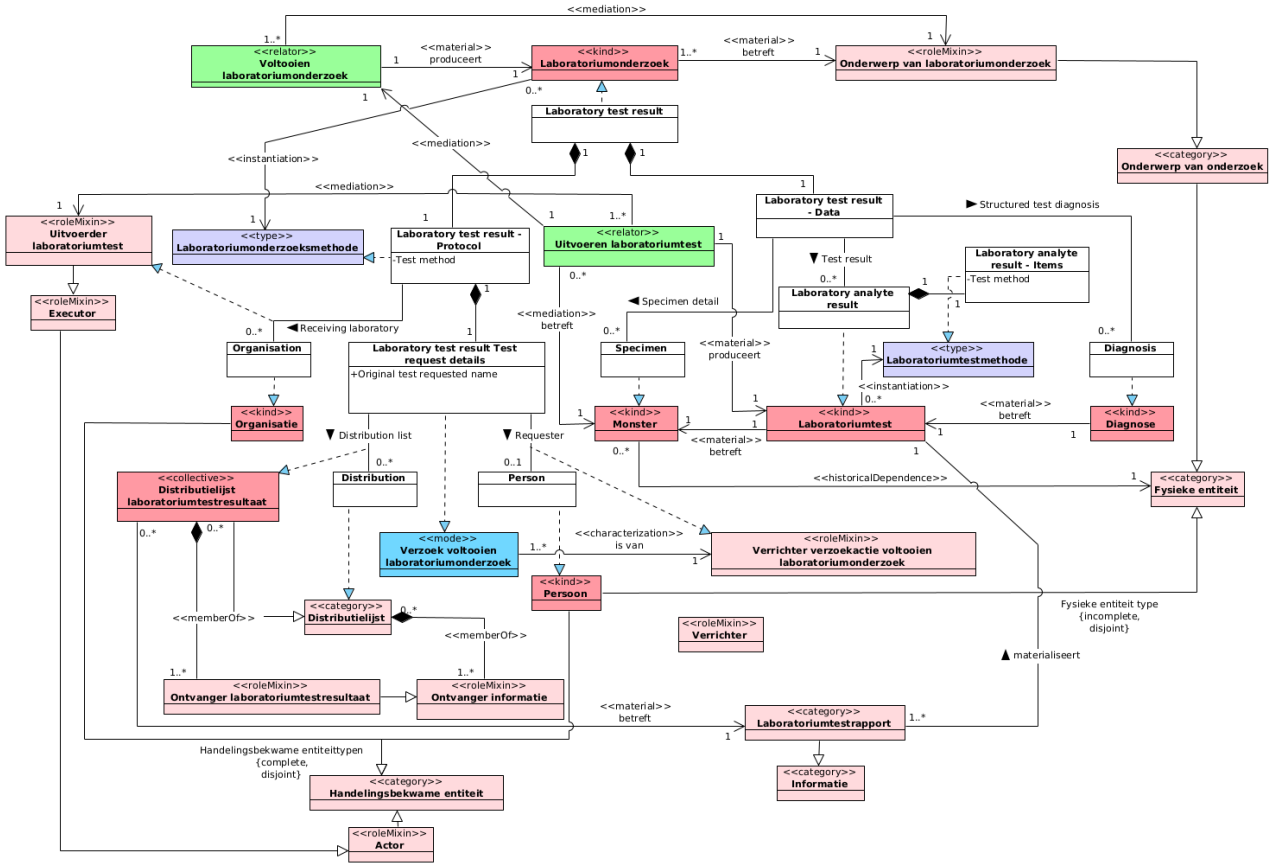
Definitie: "*Details about the transportation process of an identified item between two or more sites.*"

Link: https://ckm.openehr.org/ckm/archetypes/1013.1.3850

7.2 Mapping van formele specificaties naar openEHR

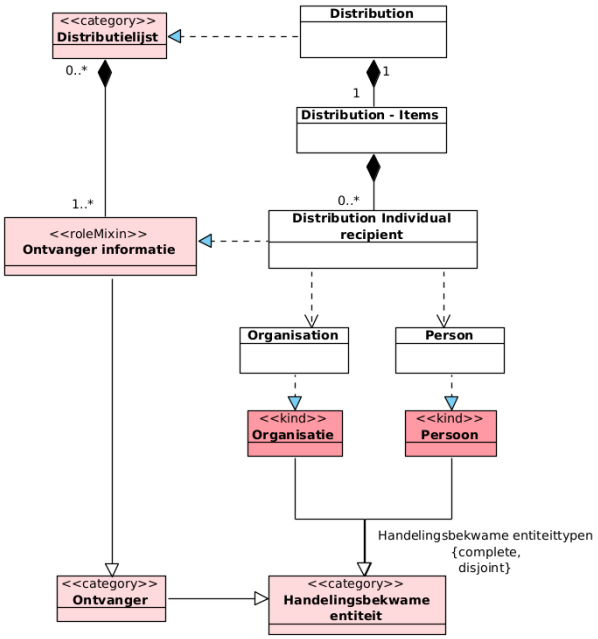
In deze paragraaf worden per openEHR archetype een mapping gegeven naar de relevante klassen uit de formele specificaties van hoofdstuk 6. Middels een gestreepte pijl met een blauwe punt, wordt deze mapping gevisualiseerd.

7.2.1 Laboratory test result



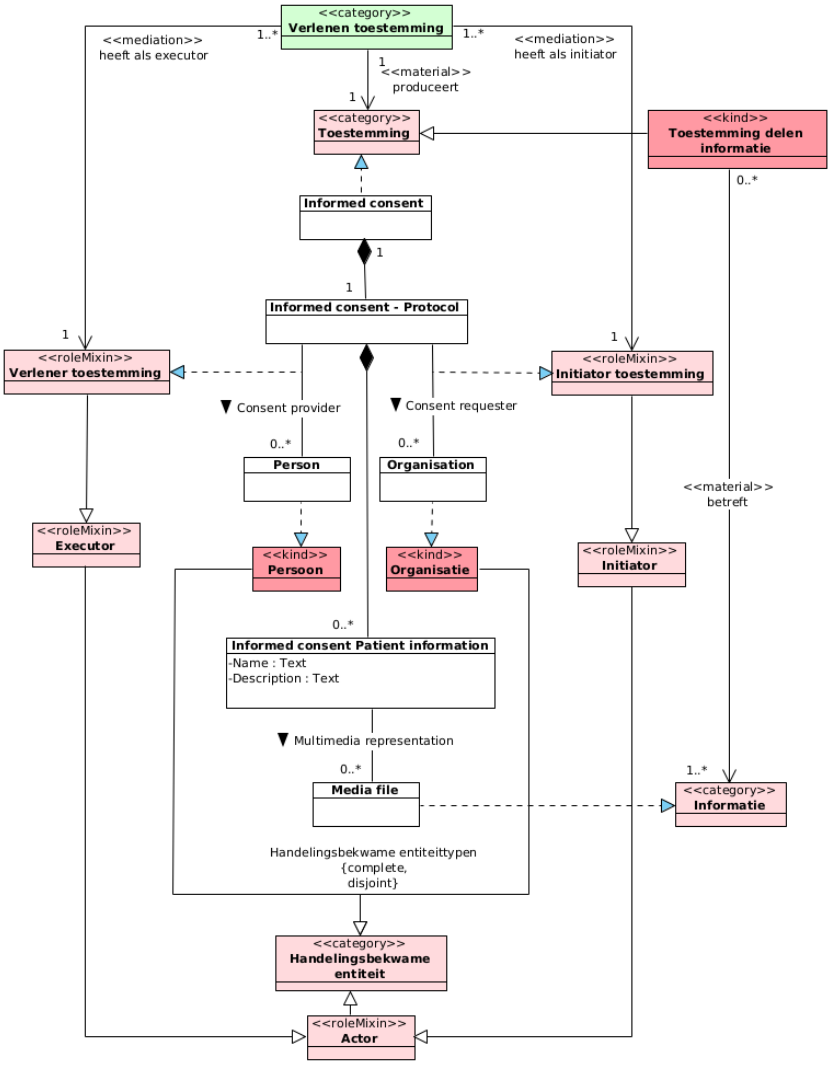
Figuur 51: Mapping naar openEHR Laboratory test result

7.2.2 Distribution



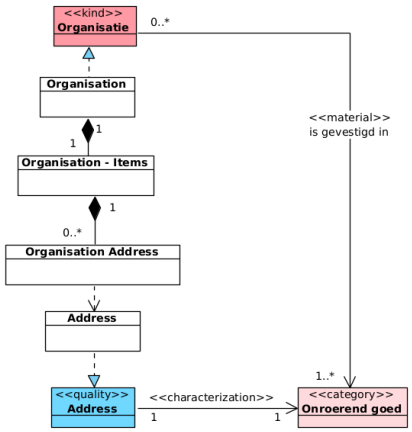
Figuur 52: Mapping naar openEHR Distribution

7.2.3 Informed consent



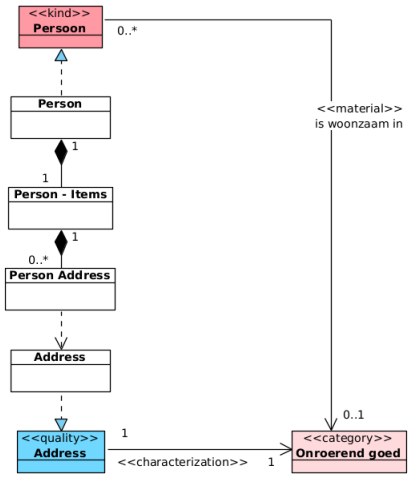
Figuur 53: Mapping naar openEHR Informed consentLaboratory test result

7.2.4 Organisation



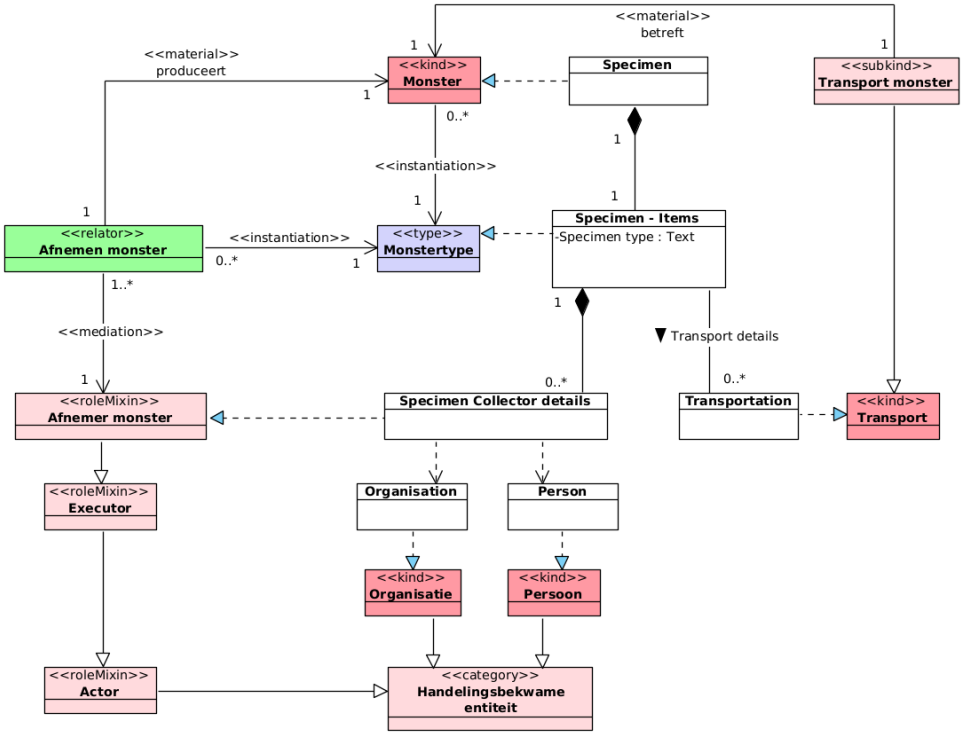
Figuur 54: Mapping naar openEHR Organisation

7.2.5 Person



Figuur 55: Mapping naar openEHR Person

7.2.6 Specimen



Figuur 56: Mapping naar openEHR Specimen

7.2.7 Transportation (of an item)

Figuur 57: Mapping naar openEHR Transportation (of an item)

8 Openstaande punten

Dit document heeft als doel om uit te leggen hoe het domeinmodel voor IZB tot stand komt en op welke manier deze, met name formeel, wordt gespecificeerd. De uitwerkingen in paragraaf 6.7 en paragraaf 7.2 zijn nog niet op compleetheid en juistheid gevalideerd. Oftewel, er moet nog het nodige gebeuren om te komen tot de volledige en correcte uitwerking van het gehele IZB-domeinmodel.

Dit hoofdstuk geeft een opsomming van de nog openstaande punten die moeten worden uitgevoerd om tot een compleet en correct IZB-domeinmodel te komen zoals die in de komende periode zullen worden uitgewerkt:

- Het Diagnostiek sub-domein moet worden aangevuld en gevalideerd om de compleetheid en juistheid te kunnen garanderen.
- De overige onderdelen van het IZB-domeinmodel moeten op dezelfde wijze worden uitgewerkt, zoals weergegeven in dit document.
- De metamodellen zoals beschreven in paragraaf 6.5 moeten nog door een externe DEMO-expert worden gevalideerd.
- De OntoUML en DEMO talen hebben een fundamenteel verschillende kijk op het concept feit, wat tot een uitdaging leidde bij het uitwerken van de DEMO- begrippen: productiefeit en coördinatiefeit. Hiervoor is een pragmatische oplossing gekozen in de huidige uitwerking van paragraaf 6.5. Er moet nader onderzoek worden gedaan naar de fundamenteel correcte wijze van het representeren van beide feitconcepten in de formele modellen.
- Het ‘beschrijvende’ perspectief op het ‘informatie’ onderdeel van het domeinmodel is niet weergegeven in dit document. Er moet een goede invulling hiervan worden toegevoegd in de ArchiMate modellen, gebruikmakend van onder andere het *Bedrijfsobject* element uit de ArchiMate taal.

Zwarte Woud 2
3524 SJ Utrecht
ggdghor.nl