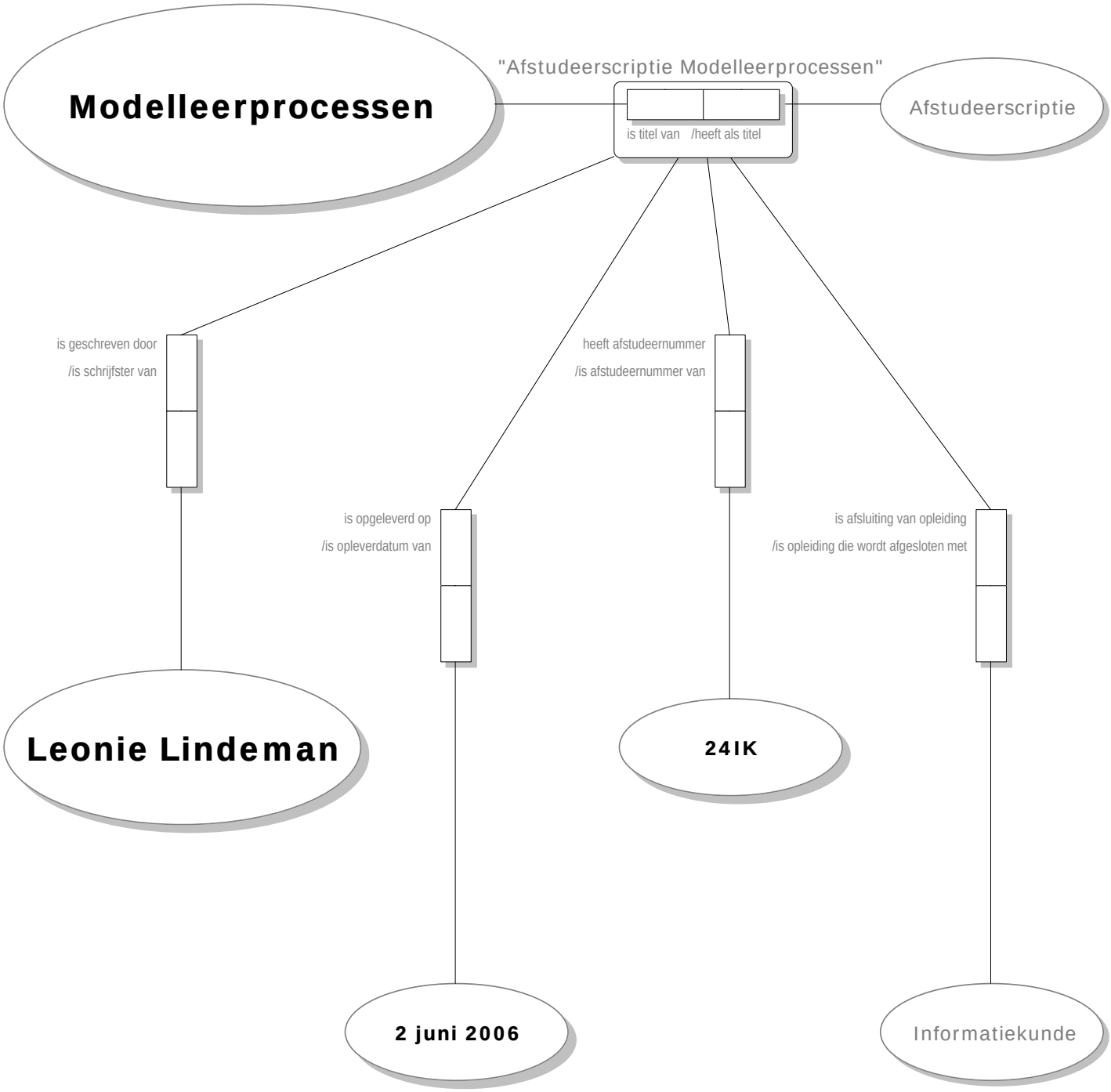

Modelleerprocessen
"Afstudeerscriptie Modelleerprocessen"
Afstudeerscriptie
is titel van
/heeft als titel
is geschreven door
/is schrijfster van
heeft afstudeernummer
/is afstudeernummer van
is opgeleverd op
/is opleverdatum van
is afsluiting van opleiding
/is opleiding die wordt afgesloten met
Leonie Lindeman
24IK
2 juni 2006
Informatiekunde

Afstudeerscriptie Informatiekunde

Modelleerprocessen

Auteur:	Leonie Lindeman
Studentnummer:	s0432245
Afstudeernummer:	24IK
Universiteit:	Radboud Universiteit Nijmegen
Faculteit:	Faculteit der Natuurwetenschappen, Wiskunde en Informatica
Instituut:	Nijmeegs Instituut voor Informatica en Informatiekunde
Begeleider:	Prof. dr. H.A. Proper
Referent:	Dr. S.J.B.A. Hoppenbrouwers
Datum:	2 juni 2006

Radboud Universiteit Nijmegen

Samenvatting

Het in deze scriptie besproken afstudeeronderzoek heeft als onderwerp "modelleerprocessen". In literatuur over systeemontwikkeling is veel over syntax, semantiek en kwaliteit van modellen en –modelleertechnieken te vinden, maar zeer weinig over het proces om tot een model te komen. Dit proces wordt daarom in dit afstudeeronderzoek onderzocht, waarbij de volgende probleemstelling centraal staat:

"Hoe zien het ORM model en het procesmodel eruit waarmee modelleerprocessen en hun verloop vastgelegd kunnen worden?"

Naar aanleiding van literatuuronderzoek zijn bovengenoemd ORM model en procesmodel opgesteld. Deze modellen zijn aan de hand van zowel literatuur als interviews gevalideerd. De scriptie is procesgericht opgesteld, elke stap die genomen is om tot de modellen te komen of deze te valideren, wordt beschreven.

Binnen het literatuuronderzoek zijn drie werkwijzen met elkaar vergeleken. Deze werkwijzen kunnen gezien worden als procedure voor het opstellen van een ORM model, een UML klassendiagram en een ER diagram. Deze werkwijzen zijn gegeneraliseerd tot een soort standaardwerkwijze, bestaande uit de volgende stappen:

- Het doel van het model bepalen;
- De modelleertechniek kiezen;
- Informatie verzamelen over het UoD, bijvoorbeeld een probleembeschrijving.
- Domeinscan uitvoeren om gevoel te krijgen voor scope en complexiteit van het domein en het identificeren van stakeholders;
- Het houden van een of meerdere modelleersessies waar vragen en antwoorden uitgewisseld worden om het domein helder te krijgen;
- Het afleiden van concepten, relaties en constraints;
- Het uitvoeren van controles op bijvoorbeeld redundantie en consistentie en het valideren van het model.

Vanuit deze standaardwerkwijze zijn een eerste versie van het ORM model en het procesmodel opgesteld. Het ORM model is in eerste instantie gevalideerd aan de hand van voorbeeld modellen die als populatie zijn toegevoegd.
Vervolgens zijn er diverse interviews gehouden met modelleerexperts om beide modellen te valideren. Uit deze interviews zijn enkele zaken naar voren gekomen die toegevoegd zijn aan de modellen, zoals:

- Het identificeren van de opdrachtgever;
- Het parkeren van een issue wanneer er discussie ontstaat binnen een modelleersessie. Er worden scenario's bedacht, mogelijke oplossingen, waar de opdrachtgever een knoop over moet doorhakken;
- Het vaststellen van een ambitieniveau, omdat oplossingen beperkt kunnen worden door bijvoorbeeld gebrek aan tijd en geld;
- Het opstellen van documentatie.

Ook zijn er nog enkele voorbeeld modelleerprocessen opgesteld ter controle van de modellen. Deze hebben geleid tot enkele aanpassingen in de notatie, maar niet tot inhoudelijke aanpassingen.

Voorwoord

In het kader van de afronding van mijn studie Informatiekunde aan de Radboud Universiteit Nijmegen (RU) heb ik een onderzoek uitgevoerd binnen het Nijmeegs Instituut voor Informatica en Informatiekunde (NIII), dat onderdeel is van de faculteit der Natuurwetenschappen, Wiskunde en Informatica (FNWI).
Deze scriptie is het resultaat van het onderzoek, dat als onderwerp “modelleerprocessen” had.

Ik wil via deze weg alle mensen bedanken die mij geholpen hebben dit resultaat te bereiken. In het bijzonder wil ik dhr. prof. dr. H.A. (Erik) Proper van de afdeling Information Retrieval and Information Systems (IRIS) bedanken voor het begeleiden van het afstudeerproject en het geven van inhoudelijke feedback. Ook dhr. dr. S.J.B.A. (Stijn) Hoppenbrouwers wil ik bedanken voor zijn feedback.
Als laatste wil ik dhr. De Vries, mevr. Bleeker en dhr. Crompvoets bedanken voor het beschikbaar stellen van hun tijd voor interviews, zodat het voor mij mogelijk was, met behulp van hun inzichten, de opgestelde modellen te valideren.

Leonie Lindeman,
Nijmegen, 2 juni 2006

Inhoudsopgave

1. Inleiding

Hoe verloopt een modelleerproces? In de wereld van systeemontwikkeling is er een keur aan modelleertalen en –technieken te vinden. In literatuur hierover wordt veel aandacht besteed aan onder andere de syntax, semantiek en kwaliteit van modellen en modelleertechnieken. Echter, aan het proces om tot een model te komen, wordt weinig aandacht besteed. Er is weinig inzicht hoe een modelleerproces daadwerkelijk verloopt. Het probleemgebied van dit afstudeeronderzoek is dan ook modelleerprocessen.

Dit inzicht is van groot belang om modelleerprocessen in de toekomst te kunnen sturen en te verbeteren. In deze afstudeerscriptie zal geprobeerd worden om dit gebrek aan inzicht op te lossen, door modelleerprocessen te bestuderen en vast te leggen. Het modelleerproces zal vastgelegd worden met behulp van een ORM model (Object Role Modeling) en een procesmodel.

Bij het opstellen van deze modellen zal gekeken worden hoe het gehele modelleerproces verloopt, welke aspecten een rol spelen bij een modelleerproces. Het gaat hierbij om modelleerprocessen zoals deze daadwerkelijk, in de praktijk, plaatsvinden. Hoe de modelleerprocessen in de meest ideale situatie zouden moeten plaatsvinden, blijft buiten beschouwing.

Dit onderzoek naar modelleerprocessen zal aansluiten bij het lopende onderzoek van de Radboud Universiteit op dit gebied [HPR05, HPW05a, HPW05b, HPW05c, HPW05d, PVH05, PW05, VHP03]. In deze scriptie zal dezelfde terminologie worden gebruikt als gebruikt wordt bij het hierboven genoemde onderzoek, tenzij anders aangegeven.

1.1 Achtergrond en begrippen

Ervaren modelleurs herkennen vaak overeenkomsten tussen nieuwe applicaties en applicaties die zij reeds eerder hebben ontworpen. Wanneer deze modelleurs patronen uit de eerdere modellen hergebruiken, kan dit leiden tot enorme besparingen in het ontwerpproces. Door het abstraheren van gelijke, specifieke concepten tot meer algemene concepten, is het eenvoudiger om een nieuwe applicatie te herkennen als een die gerelateerd is aan eerdere [Hal].
Het hergebruiken van delen van modellen wordt in de praktijk veel gebruikt. Het hergebruiken van de processen om tot die modellen te komen, komt minder vaak voor. Wanneer een modelleerproces vastgelegd wordt, kan dit een hulpmiddel zijn bij latere processen, niet alleen voor ervaren modelleurs, maar ook voor minder ervaren modelleurs, omdat zij op deze manier de eerdere processen kunnen gebruiken bij het huidige modelleerproces.

Om helder te krijgen wat er in deze scriptie verstaan wordt onder modelleren, zal allereerst het begrip ‘model’ duidelijk moeten zijn. Dit zal beschreven worden aan de hand van een activiteit, namelijk het observeren van een domein door een observator.

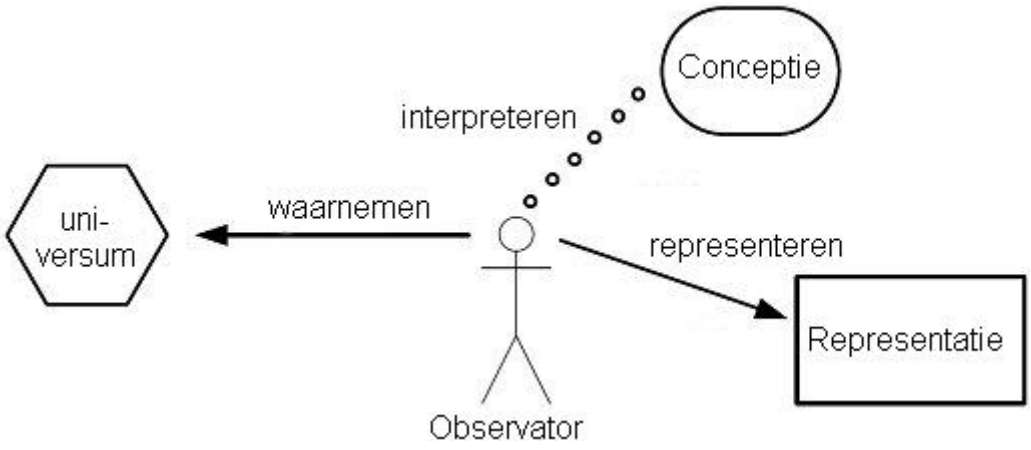
Een observator neemt een universum waar, wat leidt tot een perceptie van dit universum, en produceert vervolgens een conceptie van dat deel dat voor hem relevant is. De concepties die een observator gecreëerd heeft, zijn onmogelijk te communiceren naar andere observatoren, tenzij ze op een of andere manier gerepresenteerd zijn. Zowel de perceptie als conceptie van een observator zijn sterk beïnvloed door zijn eigen interesses in het waargenomen universum [PW05]. Ook de ervaringen die de observator reeds heeft opgedaan binnen het waargenomen universum, spelen hierbij een rol.

Uit de beschreven activiteit kunnen de volgende definities afgeleid worden, waaronder de definitie van ‘model’ [PW05]:

Observator:	Een persoon die een perceptie en conceptie van het universum produceert, gebruik makend van zijn zintuigen.
Universum:	De ‘wereld’ om de observator heen.
Perceptie:	Hetgeen dat resulteert, in het hoofd van een observator, wanneer deze het universum observeert, gebruik makend van zijn zintuigen.
Conceptie:	Hetgeen dat resulteert, in het hoofd van een observator, wanneer deze een perceptie van het universum interpreteert.
Interessegebied:	Een deel of aspect van een conceptie van het universum, waar een observator op in kan zoomen.
Model:	Een met opzet vereenvoudigde en ondubbelzinnige conceptie van een interessegebied.
Representatie:	Het resultaat van een observator die een conceptie representeert, gebruik makend van een bepaalde taal om zichzelf mee uit te drukken.

Figuur 1.1: Definities omtrent het observeren van een universum

In onderstaande illustratie is in beeld gebracht hoe de bovenstaande begrippen samenhangen en leiden tot een conceptie van een universum [PW05] .



Figuur 1.2: Het observeren van een universum

Nu duidelijk is wat een model is, en hoe een model ontstaat, kan er een definitie van ‘modelleren’ worden gegeven [PW05]:

Modelleren:	De handeling van het met opzet vereenvoudigen van (wat is aangenomen als zijnde) een deel van het universum tot een model, en het representeren van het resulterende model met behulp van een bepaalde taal en een bepaald medium.

Figuur 1.3: Definitie ‘modelleren’

Wanneer er in deze scriptie gesproken wordt over ‘modelleren’, dan wordt hiermee informatiemodellering, domeinmodellering of conceptual modeling bedoeld. Deze soorten van modelleren kennen een grote overlap, en de benamingen worden vaak door elkaar gebruikt. Deze termen maken de bovenstaande definitie van modelleren voor deze scriptie meer specifiek. Zij duiden het proces aan van het creëren van een formeel model van de concepten en relaties die bestaan in een bepaald interessegebied [BW].

Naast de in de definitie genoemde taal, omvat een modelleermethode vaak ook een procedure. Deze leidt de modelleurs in het gebruiken van de taal om op die manier modellen te produceren. Deze procedure kan gezien worden als de werkwijze om tot het model te komen en wordt vaak het modelleerproces genoemd [Hal].

1.2 Onderzoeksvragen en probleemstelling

Er is reeds enig inzicht verschaft wat betreft het centrale probleem in deze afstudeerscriptie. Om nog meer concreet te maken wat er precies onderzocht gaat worden, zijn de volgende onderzoeksvragen opgesteld, die in de scriptie beantwoord zullen worden:

O1:	Hoe ziet het ORM model eruit waarmee modelleerprocessen beschreven en vastgelegd kunnen worden?
O2:	Hoe ziet het procesmodel eruit waarmee het verloop van modelleerprocessen vastgelegd kan worden?
O3a:	Uit welke activiteiten bestaat een modelleerproces?
O3b:	Welke rol spelen deze activiteiten binnen het modelleerproces?
O3c:	In welke volgorde worden deze activiteiten uitgevoerd?
O4a:	Uit welke elementen bestaat een model?
O4b:	Hoe komen deze elementen voort uit het modelleerproces?

Figuur 1.4: Onderzoeksvragen

De eerste twee onderzoeksvragen zijn de twee belangrijkste vragen, die zullen leiden tot het uiteindelijke resultaat van de scriptie, te weten een ORM model en een procesmodel. Onderzoeksvragen O3 en O4, die verdeeld zijn in subvragen, zijn hulpmiddelen om tot de twee modellen te komen. De onderzoeksvragen kunnen samengevat worden tot één probleemstelling, de centrale vraag van het onderzoek.
Deze probleemstelling luidt als volgt:

> Hoe zien het ORM model en het procesmodel eruit waarmee modelleerprocessen en hun verloop vastgelegd kunnen worden?

Figuur 1.5: Probleemstelling

1.3 Methode

Het afstudeeronderzoek is gestart met een literatuurstudie. Na het bestuderen van een groot deel van de literatuur is een eerste versie van beide modellen opgesteld. Deze modellen zijn gecontroleerd en aangepast aan de hand van interviews met modelleerexperts. Ook de overige literatuur is gebruikt om de modellen te valideren en bij te stellen, om zo tot de definitieve versie van de modellen te komen.

Het onderzoek wordt procesgericht gepresenteerd in deze scriptie. Met andere woorden, alle stappen die in het onderzoek worden genomen, worden in een chronologische volgorde beschreven, waardoor duidelijk naar voren komt hoe de modellen tot stand zijn gekomen.

2. Literatuurstudie

Zoals reeds in de inleiding is beschreven, vormt literatuurstudie de basis van het afstudeeronderzoek. Aan de hand van diverse bronnen zal een eerste versie van de modellen gecreëerd worden.

In de literatuur zijn diverse werkwijzen te vinden voor verschillende vormen van modelleren. Deze werkwijzen zijn een leidraad bij het creëren van het model. Belangrijk hierbij is dat het niet de enige mogelijke manier is om tot een correct model te komen. Het betreft hier slechts veel voorkomende en door modelleurs als prettig ervaren werkwijzen. Uit deze werkwijzen kan afgeleid worden hoe modelleerprocessen ongeveer verlopen en uit welke activiteiten een modelleerproces bestaat. Daarnaast zullen de werkwijzen informatie verschaffen over de elementen van de modellen en welke plaats deze hebben in het modelleerproces. Deze informatie is goed bruikbaar voor het beantwoorden van de onderzoeksvragen.

De werkwijzen zullen hieronder kort beschreven worden en vervolgens worden gegeneraliseerd om tot de modellen te komen. Eventuele informatie die niet uit de werkwijzen is af te leiden, maar wel voorkomt in de literatuur en die van belang is bij het modelleerproces, zal eveneens in de modellen worden opgenomen.

De werkwijzen die vergeleken zullen worden, betreffen in de literatuur gevonden werkwijzen voor *Object Role Modeling* (ORM), *Unified Modeling Language* (UML) klassendiagrammen en *Entity Relationship* (ER) diagrammen. Onderstaande zal geen overzicht van mogelijke modelleertechnieken zijn, noch een vergelijking of afweging tussen de verschillende benaderingen. Alle drie zijn populaire, veelgebruikte notaties en kennen hun eigen sterke punten en zwakheden, die hier buiten beschouwing gelaten zullen worden.

De uiteindelijke modellen zullen getest worden aan de hand van zowel literatuur als interviews met modelleerexperts.

De in de werkwijzen gebruikte termen kunt u terugvinden in Appendix A, waar deze per benadering uitgelegd worden.

2.1 ORM

Object Role Modeling, in de rest van deze scriptie ORM genoemd, is een feit georiënteerde methode die met name wordt gebruikt voor het modelleren van een informatiesysteem op conceptueel niveau. ORM startte in begin jaren zeventig als semantische modelleerbenadering die de wereld ziet in termen van objecten die rollen spelen [Hal]. De methode maakt gebruik van natuurlijke taal en diagrammen waaraan een populatie kan worden toegevoegd met behulp van voorbeelden. Het gebruik van natuurlijke taal vertaalt zich in het weergeven van informatie in termen van elementaire zinnen.

Terry Halpin geeft in zijn boek [Hal] een *Conceptual Schema Design Procedure* (CSDP), bestaande uit zeven stappen, die helpt bij het opstellen van ORM-diagrammen. De procedure is weergegeven op de volgende bladzijde, gevolgd door een korte toelichting.

1. Transformeer informatievoorbeelden naar elementaire feiten, en pas kwaliteitscontroles toe.
2. Teken de feittypen, en voer een populatie check uit.
3. Controleer op entiteittypen die gecombineerd zouden kunnen worden, en let op rekenkundige afleidingen.
4. Voeg uniciteit constraints toe en controleer de ariteit van de feittypen.
5. Voeg verplichte rol constraints toe en controleer op logische afleidingen.
6. Voeg waarde, set vergelijking en subtype constraints toe.
7. Voeg overige constraints toe en voer laatste controles uit.

Figuur 2.1: ORM Conceptual Schema Design Procedure

Stap 1: Transformeer informatievoorbeelden naar elementaire feiten, en pas kwaliteitscontroles toe

De eerste stap van de procedure is het verzamelen van informatie over het UoD. Het kan voorkomen dat er reeds informatievoorbeelden beschikbaar zijn, bijvoorbeeld uitvoerrapporten en invoerformulieren van een reeds bestaande applicatie, of tabellen, formulieren, diagrammen of tekst. Wanneer er geen voorbeelden beschikbaar zijn, worden deze in overleg met de domeinexpert opgesteld.
Wanneer de voorbeelden beschikbaar zijn, worden deze uitgedrukt in termen van elementaire feiten. Het bijvoeglijk naamwoord ‘elementair’ geeft aan dat het feit niet opgesplitst kan worden in kleinere stukjes informatie die samen dezelfde informatie weergeven als het origineel.

Om de kwaliteit van het gedane werk te controleren, worden de volgende vragen gesteld:

- Zijn de entiteiten juist geïdentificeerd?
- Kunnen de feiten opgesplitst worden in meerdere feiten, zonder informatie te verliezen?

Stap 2: Teken de feittypen, en voer een populatie check uit

De tweede stap betreft het tekenen van een conceptueel schema waarin alle feittypen staan. Dit schema illustreert alle relevante objecttypen, predikaten en verwijzingsmodes. Zodra het schema is getekend, wordt de juistheid hiervan gecontroleerd met een voorbeeldpopulatie. Elk feittype uit het diagram wordt getest door het toevoegen van minimaal één feit als populatie, en dit feit dient teruggelezen te kunnen worden in natuurlijke taal. Dit vertaalt zich in het toevoegen van een feittabel aan elk feittype en het invullen van waarden in de relevante kolommen van deze tabel. Elke kolom in de feittabel correspondeert met een specifieke rol.

Stap 3: Controleer op entiteittypen die gecombineerd zouden kunnen worden, en let op rekenkundige afleidingen

De derde stap is bedoeld om te controleren of er entiteittypen zijn die gecombineerd kunnen worden. Ook wordt er gecontroleerd of er feittypen zijn die afgeleid kunnen worden uit andere feittypen door middel van berekeningen.
Bij het controleren op mogelijke samenvoegingen van entiteittypen wordt er gekeken naar primitieve entiteittypen, niet naar subtypen. Als er entiteittypen gevonden worden die overlappen, dan moeten deze gecombineerd worden tot één entiteittype.
Het kan voorkomen dat een bepaalde waarde afgeleid kan worden uit andere waarden, door het uitvoeren van bepaalde berekeningen. Om de kans op menselijke fouten zo klein mogelijk te maken, is het beter om het systeem deze waarde te laten afleiden dan om mensen dit te laten berekenen en opslaan.

Stap 4: Voeg uniciteit constraints toe en controleerde ariteit van de feittypen

Tot nu toe was de procedure gericht op de elementaire feittypen. De rest van de procedure is vooral gericht op het specificeren van constraints.

In een conceptueel schema mag geen redundantie optreden, oftewel er mag geen elementair feit dubbel voorkomen. Om hier voor te zorgen worden uniciteit constraints toegevoegd.
Om te controleren of alle feittypen in het schema elementair zijn, oftewel niet meer splitsbaar, wordt een check uitgevoerd die gebaseerd is op de uniciteit constraints.
Om te beslissen of feittypen opgesplitst moeten worden, wordt de zogeheten n-1 regel gebruikt. Elk n-voudig feittype heeft een sleutellengte van minimaal n-1. Als deze regel niet opgaat is het feittype niet elementair en moet deze gesplitst worden.

Stap 5: Voeg verplichte rol constraints toe en controleer op logische afleidingen

In de vijfde stap wordt er gekeken naar verplichte rol constraints en logische afleidingen. Verplichte rol constraints indiceren welke rollen gespeeld móeten worden door de populatie van een objecttype, en welke optioneel zijn. Nadat de verplichte rol constraints gespecificeerd zijn, wordt er een controle uitgevoerd om te kijken of er feittypen zijn die logisch afgeleid kunnen worden van andere feittypen.
Dit kan worden bepaald met behulp van het feit dat constraints op een afleidbaar feittype normaal gesproken ook afleidbaar zijn. Ook het gebruik van tegenvoorbeelden is hierbij een goed hulpmiddel.

Stap 6: Voeg waarde, set vergelijking en subtype constraints toe

Een waarde constraint geeft aan welke waarden toegestaan zijn in een waardetype. Een waardetype wordt gedefinieerd door de set van mogelijke waarden te declareren als één of meer opsommingen of reeksen. Dit wordt weergegeven door deze naast het waardetype zelf te zetten of naast het entiteittype dat verwijst naar het waardetype.

Set vergelijking constraints beperken de manier waarop de populatie van een rol, of rolopvolging, relateert aan de populatie van een andere. Hierbij wordt onderscheid gemaakt tussen subset constraints, gelijkheid constraints en uitsluiting constraints.
Een subset constraint geeft aan dat elk object in de populatie van de subset rol ook voor moet komen in de populatie van de superset rol. De rollen moeten gespeeld worden door hetzelfde objecttype. Een gelijkheid constraint betekent dat de populatie van de betrokken rollen altijd gelijk moet zijn. Als de populaties van rollen geen waarden gemeen mogen hebben, wordt er een uitsluiting constraint toegevoegd.
Als een uitsluiting constraint en een verplichte rol constraint gecombineerd worden, is er sprake van een “exclusieve of” constraint.

Als een objecttype geclassificeerd is in een of meer specifieke types, dan worden deze gespecialiseerde types subtypes genoemd. De belangrijkste reden voor het gebruik van subtypes is om een constraint te declareren dat een specifieke rol gespeeld wordt door alleen dat subtype. Subtypes kunnen ontstaan door generalisatie (bottom up) of specialisatie (top down).

Stap 7: Voeg overige constraints toe en voer laatste controles uit

In de laatste stap van de procedure worden nog enkele constraints toegevoegd:

- Een frequentie constraint geeft aan hoe vaak een waarde in een feitkolom moet voorkomen;

- Een ring constraint is van toepassing op rollen die een ring vormen. Als twee rollen in een predikaat gespeeld worden door hetzelfde objecttype, dan vormt het pad van het objecttype door het rolpaar en terug naar het objecttype een ring. Als de rollen gespeeld worden door subtypes met een gemeenschappelijk supertype, dan vormt het pad van en naar het supertype ook een ring;

- Een object cardinaliteit constraint limiteert de cardinaliteit van elke populatie van een objecttype;

- Een relatieve sluiting constraint geeft aan hoe compleet de kennis over optionele informatie is. Als deze constraint wordt geplaatst, wordt daarmee aangegeven dat als een object in de populatie van het objecttype deze rol speelt in de echte wereld, dat het object deze rol dan ook speelt in het model.

Naast het toevoegen van de laatste constraints, worden er aan het eind van de procedure enkele controles uitgevoerd om de kwaliteit van het schema te controleren. Deze controles worden uitgevoerd om zeker te zijn van:

- Interne consistentie; het constraint patroon kan gepopuleerd worden zonder contradictie;

- Externe consistentie; het schema komt overeen met originele data en condities;

- Afwezigheid van redundantie; elementaire feiten kunnen niet herhaaldelijk voorkomen;

- Compleetheid; het schema dekt alle eisen.

2.2 UML klassendiagram

UML staat voor *Unified Modeling Language*. Het is een modelmatige taal die is ontworpen om objectgeoriënteerde analyses en ontwerpen voor een informatiesysteem te kunnen maken. De UML modellen zijn een grafische weergave van bepaalde aspecten van het informatiesysteem [Wiki]. UML 2.0 definieert dertien typen diagrammen, onderverdeeld in drie categorieën, te weten statische applicatiestructuur, gedrag en andere aspecten van interacties. Het UML klassendiagram valt in de categorie structuur diagrammen [OMG].

Een klasse is een verzameling van objecten die overeenkomstige eigenschappen bezitten. Deze eigenschappen worden verdeeld in attributen en operaties.

Binnen UML is geen werkwijze vastgelegd voor het UML klassendiagram. Toch zijn er wel degelijk stappenplannen beschikbaar die beschrijven hoe men te werk kan gaan om tot een klassendiagram te komen. Een voorbeeld hiervan is onderstaand stappenplan, dat wordt gevolgd door een toelichting [WK]:

1. Identificeer alle mogelijke kandidaatklassen.
2. Selecteer de klassen uit de lijst van kandidaten.
3. Maak een modeldictionary.
4. Identificeer associaties.
5. Identificeer attributen.
6. Identificeer operaties.
7. Generaliseer met behulp van overerving.
8. Maak OCL-constraints.
9. Groepeer klassen eventueel in packages.
10. Itereer over de gedane stappen.

Figuur 2.2: Werkwijze UML klassendiagram

Stap 1: Identificeer alle mogelijke kandidaatklassen

Het doel van deze eerste stap is het vinden van zoveel mogelijk kandidaatklassen. Dit kan gedaan worden door alle zelfstandige naamwoorden uit de probleembeschrijving te onderstrepen, indien deze aanwezig is. Indien er vooraf use-cases zijn gemaakt, kunnen ook hier de zelfstandige naamwoorden uit worden meegenomen. Een betere manier is echter om brainstormsessies te houden, waarbij zowel de toekomstige gebruikers, domeinexperts als ontwikkelaars aanwezig zijn. Bij deze sessies worden de stappen één tot en met drie van het stappenplan uitgevoerd. Het resultaat is een grote lijst van kandidaatklassen. Daarnaast krijgen alle betrokkenen inzicht in de manier waarop de anderen het systeem benaderen.

Stap 2: Selecteer de klassen uit de lijst van kandidaten

De volgende stap is het selecteren van klassen van de lijst die in de eerste stap is samengesteld. Voor iedere term wordt bepaald of deze wel of niet als klasse in het systeem opgenomen zal worden. Van iedere term wordt een definitie of beschrijving gemaakt. Aan de hand van een paar vragen wordt bepaald of de klasse in het systeem zal worden opgenomen of niet.

De volgende vragen zijn van belang:

- Speelt de kandidaatklasse een zelfstandige rol in het probleemdomein?
- Willen we iets van de kandidaat weten of willen we dat hij iets voor ons doet?
- Heeft de kandidaatklasse een duidelijke verantwoordelijkheid in het systeem?

Wanneer bovenstaande vragen met “ja” beantwoord worden, dan wordt de kandidaat een klasse.

Een andere manier om de klassen te selecteren is het kijken naar waarom een kandidaat géén klasse zou zijn, bijvoorbeeld de klasse is redundant, de klasse is irrelevant of komt in aanmerking als attribuut of associatie.

Stap 3: Maak een modeldictionary

De derde stap in de werkwijze is het opstellen van een modeldictionary. Hierin worden alle gevonden termen opgenomen, waarvan de beschrijving al grotendeels is gemaakt in de vorige stap. Eventueel kunnen termen die niet in het klassendiagram terug zullen komen, opgenomen worden in een apart deel van de modeldictionary, dat kan dienen als referentiekader.

Stap 4: Identificeer associaties

In stap vier wordt er geprobeerd associaties tussen klassen te vinden, oftewel paren van klassen die iets, maakt niet uit wat, met elkaar te maken hebben. Het meest praktisch is om alle paren van klassen te bekijken en alle klassen die iets met elkaar te maken hebben, op te schrijven. Hierdoor ontstaat een lijst met kandidaatasscociaties. Vervolgens moet er gekeken worden welke kandidaten in aanmerking komen als associatie in het model. Kandidaatassociaties die worden afgewezen, zijn bijvoorbeeld:

- irrelevante associaties;
- acties (een associatie is een structurele relatie, een actie is slechts tijdelijk);
- afgeleide associaties.

Stap 5: Identificeer attributen

Nadat de associaties zijn geïdentificeerd, wordt in de vijfde stap het klassendiagram verder ingevuld, door het toevoegen van attributen. Een aantal attributen is uit de lijst van stap één te halen, wanneer deze werd afgewezen als klasse, omdat dit een attribuut betrof. Deze kunnen nu toegevoegd worden aan één van de objecten. Daarnaast kunnen attributen afgeleid worden uit de beschrijving van de klassen in de modeldictionary.

Bij ieder attribuut moet afgevraagd worden of dit wel degelijk een attribuut is, bijvoorbeeld door het stellen van de volgende vragen:

- Wat zijn de dingen waar een object van weet?
- Wat voor informatie zouden we aan het object kunnen vragen?

Bij het erkennen van attributen moet op het volgende gelet worden:

- Mogelijke attributen mogen niet verwijzen naar een object, aangezien het dan een associatie betreft;
- Mogelijke attributen mogen geen identifiers zijn;
- De attributen mogen niet alleen intern in het object relevantie hebben;
- Er mogen niet teveel gedetailleerde attributen worden toegevoegd aan de klassen.

Stap 6: Identificeer operaties

De zesde stap betekent het op zoek gaan naar operaties bij de objecten. Een aantal operaties is te halen uit de lijst van kandidaatklassen uit de eerste stap. De belangrijkste vragen die we bij ieder object stellen, zijn:

- Wat wil ik dat het object voor mij doet?
- Wat is de verantwoordelijkheid van dit object en hoe kan het deze vervullen?

Veel operaties komen pas in een later stadium naar boven; het is niet erg als tijdens de domeinanalyse de operaties nog niet compleet zijn. Het is de bedoeling dat er een goed inzicht ontstaat in de verantwoordelijkheden van de klassen en de rol hiervan in het geheel. Teveel operaties per klasse maken het diagram onoverzichtelijk.

Stap 7: Generaliseer met behulp van overerving

Na de eerste zes stappen is er een eerste versie van het klassendiagram beschikbaar dat bestaat uit klassen met attributen en operaties, en associaties tussen de klassen. Deze eerste versie wordt in de zevende stap bestudeerd, om te kijken of er generalisaties gevonden kunnen worden van de bestaande klassen. Op deze manier kunnen er nieuwe (super)klassen gevonden worden.

Generaliseren is het vinden van overeenkomsten in verschillende klassen. Het gaat hier om overeenkomsten tussen de operaties en/of attributen die in de vorige stappen bij de klassen gedefinieerd zijn. Wanneer meerdere klassen een aantal identieke operaties en/of attributen hebben, dan kan voor deze klassen een superklasse gedefinieerd worden. De identieke operaties en attributen worden uit de oorspronkelijke klassen gehaald en verplaatst naar de nieuwe superklasse.

Naast het controleren op identieke operaties en attributen, kan ook gekeken worden naar identieke associaties. Klassen kunnen gegeneraliseerd worden wanneer deze identieke associaties hebben. De associaties uit de subklassen worden in dit geval vervangen door één associatie van de superklasse.

Stap 8: Maak OCL-constraints

De achtste stap van deze werkwijze is bedoeld om het klassendiagram meer precies te maken door het op een exacte manier toevoegen van bedrijfsregels en beperkingen die in het domein gelden. Daarnaast wordt in deze stap gecontroleerd of met behulp van het klassendiagram voldaan kan worden aan de belangrijkste informatiebehoeften voor het systeem.

Er wordt een aantal vragen geformuleerd, die binnen het probleemdomein gesteld zouden kunnen worden. Deze vragen hoeven niet binnen de functionele eisen van het systeem te liggen en mogen ook vragen zijn die in de toekomst gesteld zouden kunnen worden. De vragen en/of antwoorden kunnen met behulp van OCL worden genoteerd. OCL staat voor *Object Constraint Language* en is een taal die het mogelijk maakt om expressies en constraints op object georiënteerde modellen te beschrijven [KO].

Wanneer zinvolle vragen niet beantwoord kunnen worden met behulp van het klassendiagram, dan betekent dit dat het model een te beperkte weergave is van het probleemdomein. Er zal dan besloten moeten worden of het model aangepast wordt, of dat er met de te beperkte weergave te leven is.

Stap 9 : Groepeer klassen eventueel in packages

Een klassendiagram heeft de neiging om te groeien, dus is het handig om het in overzichtelijke delen op te delen. Voor het maken van een opdeling zijn veel verschillende mogelijkheden. UML kent een package mechanisme, dat de gebruiker vrij laat in de manier van opdelen van het systeem. Vanuit het oogpunt van domeinanalyse is de enige zinvolle opdeling een logische opdeling die zijn oorsprong vindt in het probleemdomein.

Stap 10: Itereer over de gedane stappen

Het hierboven beschreven proces lijkt sequentieel, echter in de praktijk is dit nooit het geval. Het eerste klassendiagram dat ontwikkeld wordt, zal altijd nog een aantal fouten of onvolkomenheden bevatten. Vandaar dat er regelmatig een paar stappen teruggegaan moet worden om aanpassingen aan het diagram aan te brengen. Hierbij wordt niet gewacht tot de laatste stap, maar worden de aanpassingen direct gedaan wanneer er inconsistenties of incompleetheid wordt ontdekt.

2.3 ER diagram

Het Entity Relationship (ER) model is in 1976 geïntroduceerd door Peter Chen. Sinds die tijd is het model uitgebreid door Chen en vele anderen. Er is momenteel niet één algemeen geaccepteerd standaard ER model, maar er bestaan verscheidene overeenkomstige componenten die in de meeste varianten van het ER model voorkomen. Een ER diagram is het grafische resultaat van het ER model en geeft de entiteiten en hun onderlinge samenhang weer [Kroen].

In de literatuur zijn verschillende richtlijnen te vinden voor het opstellen van een ER diagram, zoals de werkwijze in [HSZ]. Deze gaat er vanuit dat er voordat met de werkwijze begonnen kan worden, eerst materiaal verzameld dient te worden, zoals bijvoorbeeld een document van eisen, formulieren, rapporten en andere modellen. De methode die bij de werkwijze, die hieronder weergegeven is, gebruikt wordt, is het bestuderen van feiten en/of zinnen.

1. Identificeer mogelijke entiteittypen.
2. Identificeer mogelijke relatietypen tussen de geïdentificeerde entiteittypen.
3. Bepaal de cardinaliteiten van de relatietypen.
4. Identificeer en associeer attributen met entiteittypen.
5. Identificeer en associeer attributen met relatietypen.
6. Bepaal de domeinen van de attributen.
7. Bepaal voor elk entiteittype de potentiële identifiers.
8. Overweeg het gebruik van specialisatie/generalisatie (sub-/supertypes).
9. Controleer het model op redundantie.
10. Valideer het ER-model.
11. Houd een review over het model met de gebruiker.

Figuur 2.3: Werkwijze ER diagram

Stap 1: Identificeer mogelijke entiteittypen

In de eerste stap van deze werkwijze worden mogelijke entiteittypen geïdentificeerd. Zelfstandige naamwoorden zijn mogelijke entiteittypen, maar kunnen later relatietypen of attributen blijken te zijn. Bij het aanmerken van zelfstandige naamwoorden als entiteittypen moet opgepast worden voor synoniemen en homoniemen. Niet alle zelfstandige naamwoorden komen in aanmerking als entiteittype, bijvoorbeeld wanneer ze iets vertegenwoordigen waar geen informatie over hoeft te worden bewaard of omdat ze eerder duiden op een attribuuttype. Richtlijn in dit stadium is “liever te veel entiteittypen dan te weinig”. Het is belangrijk dat de gevonden entiteittypen namen krijgen die zinvol zijn en duidelijk voor de gebruiker.

Stap 2: Identificeer mogelijke relatietypen tussen de geïdentificeerde entiteittypen

De tweede stap is bedoeld om mogelijke relatietypen te identificeren. Werkwoordsvormen die geselecteerde entiteittypen als subject èn object hebben zijn mogelijke relatietypen. De gevonden relatietypen dienen namen te krijgen die zinvol zijn en duidelijk voor de gebruiker.

Stap 3: Bepaal de cardinaliteiten van de relatietypen

In deze stap worden de cardinaliteiten van de relatietypen bepaald. Het is van belang dat ook de cardinaliteitsconstraints die niet in het ER diagram passen gedocumenteerd worden. Hieronder vallen bijvoorbeeld specifieke waarden van de cardinaliteit, minimum- of maximumwaarden. Deze constraints dienen tenslotte ook te worden geïmplementeerd.

Stap 4: Identificeer en associeer attributen met entiteittypen

In stap vier wordt er gekeken of er attributen zijn die bij een entiteittype horen. Een zelfstandig naamwoord is een attribuut als het iets beschrijft van een entiteit, zoals een eigenschap. Een hulpmiddel om attributen te onderscheiden kan de volgende vraag zijn:

- Welke informatie moet er een entiteit bewaard worden?

Bij twijfelgevallen wordt de gebruiker geraadpleegd.
De attributen worden toegevoegd aan de entiteittypen. Hierbij moet opgepast worden voor het toevoegen van afgeleide attributen.

Stap 5: Identificeer en associeer attributen met relatietypen

Naast attributen van entiteittypen moet er ook gekeken worden naar attributen van relatietypen. Het afvragen of er bepaalde informatie over het relatietype bewaard moet worden, is ook in deze stap een hulpmiddel. Als dit het geval is moet het relatietype omgezet worden in een nieuw, zwak entiteittype met nieuwe bijbehorende relatietypen. Het entiteittype is zwak in relatie tot de twee entiteittypen die bij het oorspronkelijke relatietype horen. Het entiteittype wordt ook wel een associatief entiteittype genoemd.

De attributen worden toegevoegd aan de nieuwe entiteittypen. Ook nu moet er opgepast worden voor het toevoegen van afgeleide attributen.

Stap 6: Bepaal de domeinen van de attributen

Een domein van een attribuut is de verzameling van mogelijke waarden van een attribuut. Deze dienen goed gedocumenteerd te worden. Deze stap levert soms extra constraints op die later procedureel bewaakt moeten worden.

Stap 7: Bepaal voor elk entiteittype de potentiële identifiers

Een potentiële identifier geeft aan welke minimale combinatie van attributen instanties van het entiteittype uniek identificeert. Dit levert één of meer kandidaatidentifiers op. Eén hiervan moet gekozen worden als primaire identifier. De overige kandidaten dienen goed gedocumenteerd te worden, dit worden later uniciteitconstraints.

Stap 8: Overweeg het gebruik van specialisatie/generalisatie (sub-/supertypes)

In deze stap dient onderzocht te worden of er entiteiten zijn die in aanmerking komen om opgesplitst te worden in een supertype en één of meer subtypes. Deze benadering wordt aangeduid als de specialisatiebenadering. Ook de generalisatiebenadering is van toepassing, waarbij onderzocht wordt of er twee of meer entiteittypen zijn die gemeenschappelijke kenmerken hebben, zodat een supertype van die entiteiten kan worden gedefinieerd.

Stap 9: Controleer het model op redundantie

In de stappen vier en vijf zijn de afgeleide attributen reeds verwijderd uit het model. In deze stap wordt gekeken naar relatietypen. Eén op één relatietypen dienen heroverwogen te worden. Deze kunnen zijn ontstaan als synoniemen over het hoofd gezien zijn. In zo'n geval moeten de twee entiteittypen worden samengevoegd.

Ook redundante relatietypen moeten verwijderd worden. Een relatietype is redundant als dezelfde informatie ook via een ander relatietype kan worden verkregen. Er moet dus gekeken worden of er tussen twee entiteittypen meer dan één pad bestaat. Een pad bestaat uit een aantal relatietypen en is vrij gemakkelijk te onderzoeken. Echter, het bestaan van meerdere paden betekent niet direct dat er een overbodig relatietype is. Het is daarom van belang dat zorgvuldig onderzocht wordt wat de betekenissen van de relatietypes in de verschillende paden zijn.

Stap 10: Valideer het ER-model

In deze stap wordt het model handmatig getest aan de hand van de gegeven feiten in het UoD en de vereiste gebruikerstransacties. Onderzocht wordt of alle informatie eruit te halen is zoals deze beschreven is in het document van eisen. Zo niet, dan moet er besloten worden of het model aangepast gaat worden, of dat het zo gelaten wordt.

Stap 11: Houd een review over het model met de gebruiker

De laatste stap van de werkwijze is het houden van een review over het model met de gebruiker.

2.4 Generalisatie en aanpassing werkwijzen

In de praktijk zal er voordat met het maken van een model begonnen kan worden, eerst een geldig doel geïdentificeerd moeten worden voor het creëren van het model. Volgens [Amb] zijn er twee primaire redenen om te modelleren. De eerste reden is als hulpmiddel voor de modelleur om uit te vinden wat het precies is waaraan hij aan het werken is. Bijvoorbeeld het modelleren van eisen van het systeem, of het analyseren van de eisen, of om een gedetailleerd ontwerp voor het systeem op te stellen. Met andere woorden, de modellen worden gebruikt om een beter begrip te kweken van één of meer aspecten van het systeem. De tweede reden om te modelleren is het bevorderen van de communicatie binnen het ontwikkelteam, of met individuen en/of groepen die buiten het team staan.

Nadat bepaald is wat het doel van het model is, is het van belang om het juiste soort model te kiezen [Amb]. Ieder artefact heeft zijn eigen specifieke sterke en zwakke punten. Voor elk soort model geldt dat deze in bepaalde situaties wel te gebruiken is en in andere niet.

Alle drie de werkwijzen schrijven voor dat er voordat er gemodelleerd kan worden, er eerst informatie verzameld moet worden over het UoD. Het verzamelen van informatie kan volgens deze werkwijzen op verschillende manieren gebeuren, onder andere door het bestuderen van reeds bestaande uitvoerrapporten en invoerformulieren, tabellen, formulieren, diagrammen of tekst. Wanneer er een probleembeschrijving aanwezig is of wanneer er reeds modellen gemaakt zijn, kunnen deze eveneens worden gebruikt om informatie te verzamelen. Een effectieve wijze om informatie te verzamelen is een modelleersessie of brainstormsessie. Brainstormsessies hebben over het algemeen meerdere aanwezigen, zoals in de werkwijze van het UML klassendiagram bij stap één, waar aanwezigheid van zowel de toekomstige gebruikers, domeinexperts als ontwikkelaars wordt gesuggereerd.

In [HBP] wordt de stap betreffende het verzamelen van informatie verder uitgediept. Het modelleerproces begint met een korte scan of waardering van het domein om een gevoel te krijgen voor de scope, diversiteit en complexiteit van het domein, evenals het identificeren van relevante stakeholders voor het domein. Hiervoor worden diverse soorten input documenten verzameld en bestudeerd, zodat er enig begrip van het UoD gekweekt wordt. Vervolgens worden er formele beslissingen genomen wat betreft de concepten die een rol spelen in het UoD, oftewel het bepalen van de scope. De volgende stap is het vaststellen van relevante concepten, wat bijvoorbeeld door middel van een modelleersessie kan plaatsvinden.

In [HPW05a, HPW05b, HPW05c] wordt een modelleersessie beschreven als een dialoog tussen een domeinexpert en een systeemanalist. Hoewel deze situatie vereenvoudigd is, is dit een toegankelijke manier om over een modelleersessie te redeneren.

De domeinexpert weet alles over het domein, of kan als dat nodig is, daar meer informatie over vinden. De domeinexpert kan statements over het domein genereren en valideren, maar is niet bekwaam om deze te formaliseren. De systeemanalist heeft geen kennis over het domein, maar weet hoe een verifieerbaar correct formeel model gecreëerd dient te worden [HPW05a].
Vragen en antwoorden worden uitgewisseld. Elke participant heeft zijn eigen kijk (conceptie) op het domein. De enige manier waarop de participanten hun modelleerdoelen kunnen behalen, is door met elkaar te communiceren, te onthouden wat er is gezegd en hier op voort te bouwen. Binnen een modelleerdialoog kunnen verschillende acties onderscheiden worden [HPW05b]:

Voorstellen(a,s)	Actor a stelt statement s voor. Dit statement wordt geen deel van het model totdat elke actor het geaccepteerd heeft.
Verwijderen(a,s)	Actor a verwijdert statement s. Verwijderen is het tegenovergestelde van voorstellen.
Accepteren(a,s)	Actor a accepteert statement s als geldig statement; het mag eventueel deel uit gaan maken van het model. Een statement kan pas geaccepteerd worden als het voorgesteld is.
Verwerpen(a,s)	Actor a verwerpt statement s, omdat a vindt dat s onacceptabel is. Verwerpen is het tegenovergestelde van accepteren.
Vragen(a,q)	Actor a stelt een vraag q, om beantwoord te worden door een actor. Vragen kunnen verwijderd of beantwoord worden.
Antwoorden(a,q,s)	Actor a beantwoordt vraag q met statement s. Een antwoord functioneert als een speciaal voorstel.

Figuur 2.4: Acties binnen een modelleerdialoog

De acties in figuur 2.4 kunnen omgezet worden in onderstaand voorbeeld van een modelleerdialoog. DE staat voor domeinexpert, SA voor systeemanalist.

Voorstellen	DE	Janssen werkt voor verkoop.
Vragen	SA	Wat voor ding is Janssen?
Voorstellen	DE	Janssen is een persoon.
Vragen/Voorstellen	SA	Onderscheiden we Janssen van andere personen door middel van zijn achternaam?
Accepteren	DE, SA	Ja [we onderscheiden Janssen van andere personen door middel van zijn achternaam.]
Vragen/Voorstellen	SA	Bent u het er mee eens dat Janssen een persoon is met achternaam Janssen?
Accepteren	DE,SA	Ja [Janssen is een persoon met achternaam Janssen.]

Figuur 2.5: Voorbeeld van een modelleerdialoog

Na de modelleersessie is er een verzameling van statements beschikbaar. Vanuit deze statements kunnen concepten afgeleid worden, evenals de relaties tussen de concepten en bijbehorende constraints. Het definiëren van de concepten kan ook het identificeren van regels en constraints betreffende de instanties van de concepten omvatten [PBH]. Bij de drie besproken werkwijzen wordt het identificeren van relaties en constraints pas in een later stadium uitgevoerd. Bij de generalisatie van de werkwijze zal deze splitsing blijven bestaan, zodat het onderscheid tussen het identificeren van de concepten, relaties en constraints zichtbaar blijft.

De naamgeving van de concepten is erg belangrijk. Wanneer de domeinexperts allemaal dezelfde term gebruiken voor een bepaald concept, dan dient de modelleur deze ook te gebruiken in zijn model. Het kan voorkomen dat binnen een domein verschillende termen gebruikt worden voor eenzelfde concept. In dat geval dient de modelleur de domeinexperts een standaard term te laten opstellen, en kan hij eventueel wat synoniemen in ogenschouw houden die ze willen blijven gebruiken [Hal]. Echter, het ontkennen van deze complexiteit door het vaststellen van een standaardterm is niet realistisch bij grootschalige applicaties [PBH]. Dit zal hier echter buiten beschouwing worden gelaten.
Idealiter worden de gedefinieerde concepten gedocumenteerd [PBH]. In de praktijk wordt dit echter lang niet altijd gedaan.

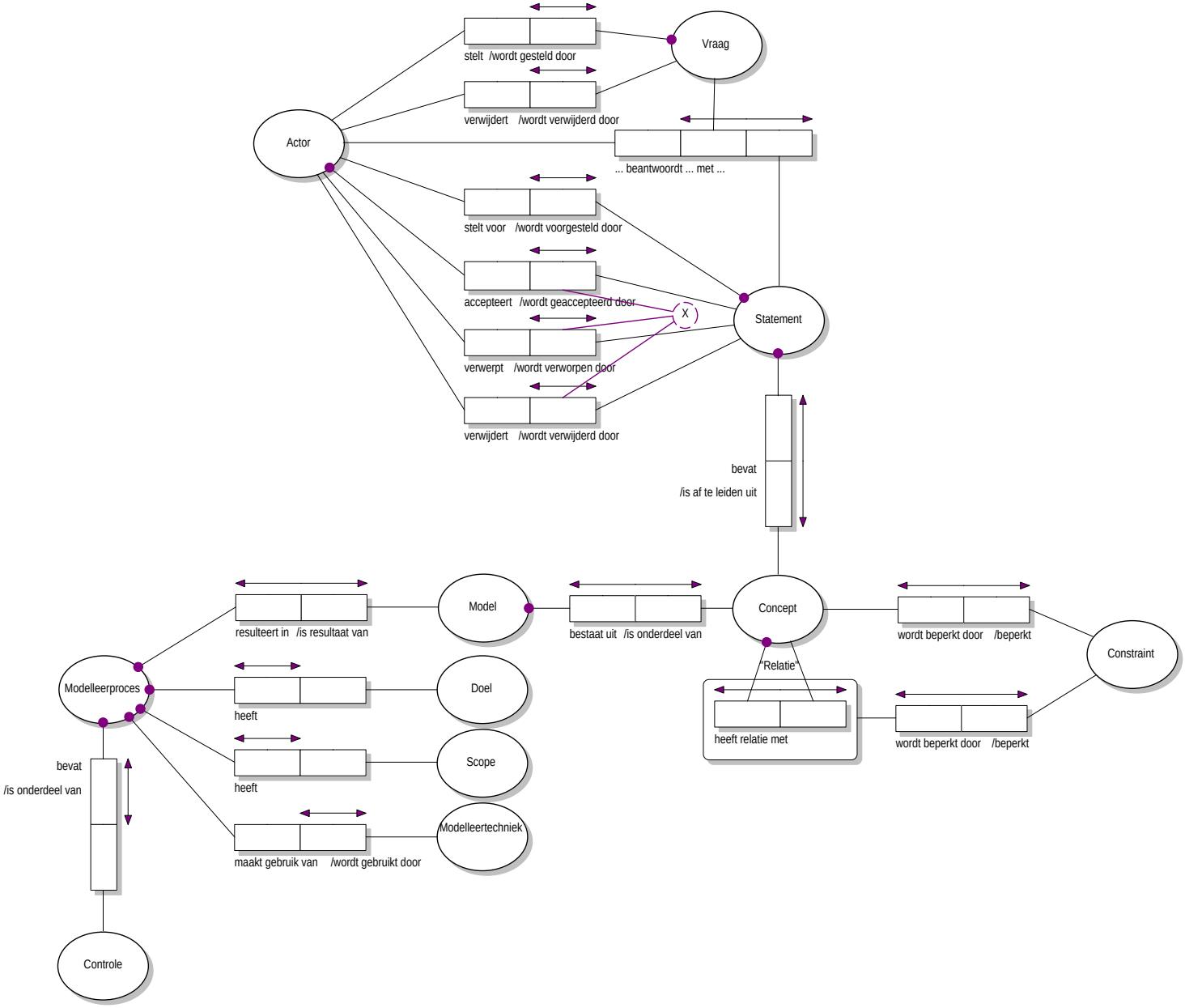
Nadat de concepten gedefinieerd zijn, kunnen de relaties tussen de concepten en de constraints vastgesteld worden. In de drie gegeneraliseerde werkwijzen wordt eerst naar de relaties gekeken, voordat de constraints geïdentificeerd worden. In de praktijk zullen deze stappen niet exact verlopen zoals beschreven [Hal], en kunnen deze stappen parallel uitgevoerd worden. Het definiëren van de constraints omvat alle mogelijke soorten constraints die bij de betreffende modelleertechniek van toepassing zijn, zoals bijvoorbeeld cardinaliteit constraints, waarden constraints en uniciteit constraints.

Voor bovenstaande stappen kan het moeilijk zijn om de mate van detaillering te bepalen. De mate van detaillering hangt van de situatie, met welk doel het model gebruikt wordt. Wanneer het model voldoende gedetailleerd is, oftewel het doel is gehaald, dan kan er gestopt worden met modelleren [Amb].

In feite is het model na de voorgaande stappen compleet. Toch is het modelleerproces nog niet ten einde, daar het model nog gecontroleerd dient te worden. Bij de besproken werkwijzen bestaan verschillende soorten checks om te controleren op onder andere redundantie, afleidingen, consistentie en compleetheid en om het model te valideren. Wanneer er een probleem gevonden wordt in het model, dan kan de modelleur er voor kiezen om het model te updaten op dat punt, of het te accepteren zoals het is. In een bepaalde situatie kan het model goed genoeg zijn, ook al is deze niet perfect. Ditzelfde geldt voor consistentie; het is niet altijd erg als een model op een punt inconsistent is, als het maar consistent genoeg is [Amb].

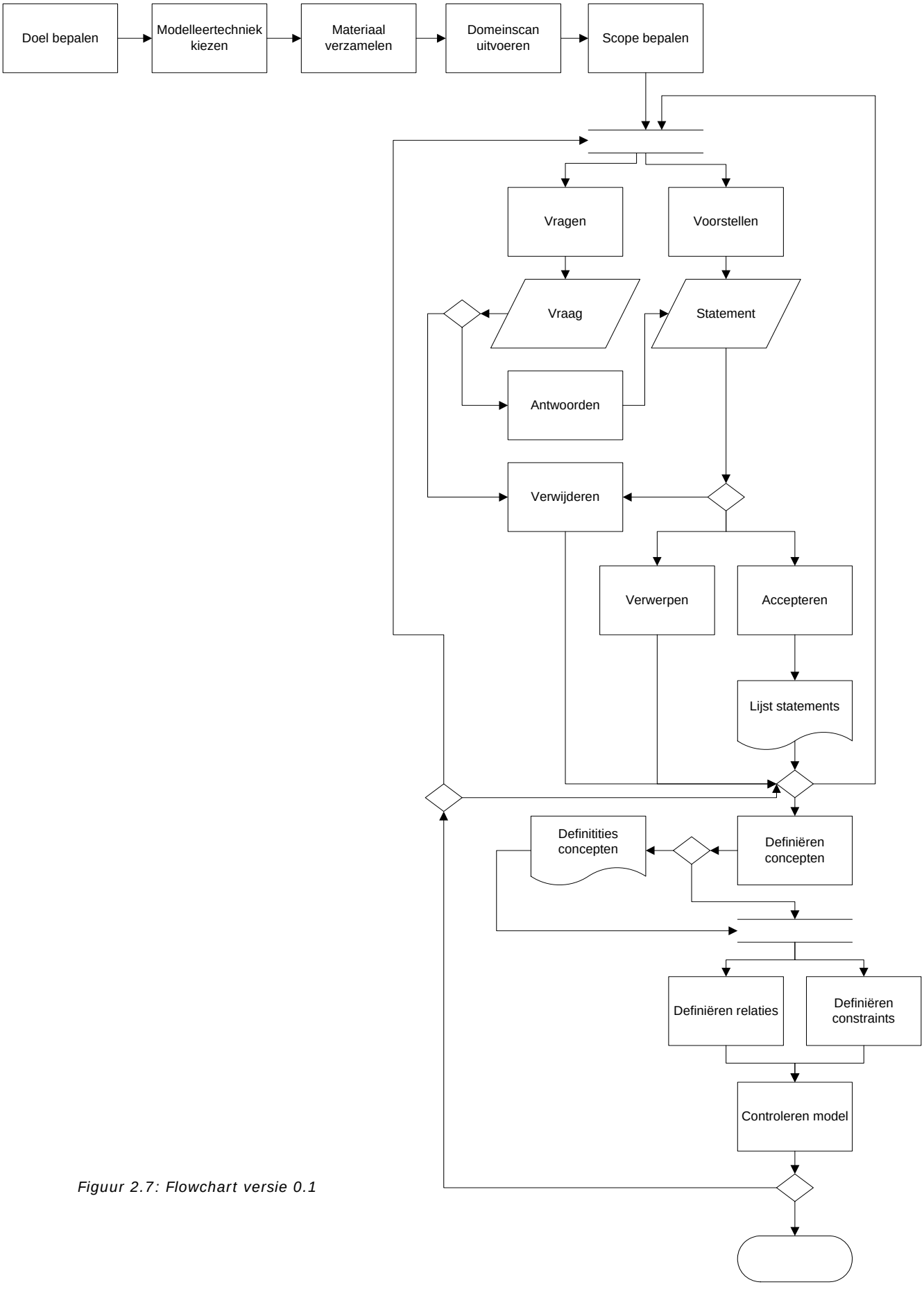
Bovenstaande generalisatie van de werkwijzen is omgezet in twee modellen, te weten een een ORM model en een flowchart. De legenda van de symbolen uit de flowchart is terug te vinden in Appendix B. De flowchart is bedoeld om de volgorde van de verschillende stappen in kaart te brengen, terwijl het ORM model de concepten en hun samenhang weergeeft.

2.5 ORM model versie 0.1



Figuur 2.6: ORM model versie 0.1

2.6 Flowchart versie 0.1

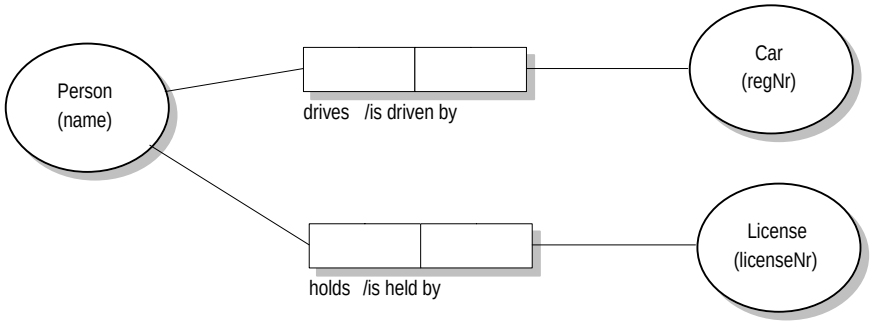


Figuur 2.7: Flowchart versie 0.1

3. Validatie van het ORM model

Aan de hand van literatuur en het toevoegen van populatie zal ORM model versie 0.1 getest worden.
Allereerst zal slechts een gedeelte van het model getest worden, te weten het gedeelte waar modellen in vastgelegd kunnen worden. Er zal begonnen worden met zeer eenvoudige modellen in verschillende notaties, waarna de toe te voegen modellen steeds uitgebreider en waarschijnlijk moeilijker toe te voegen zullen worden.

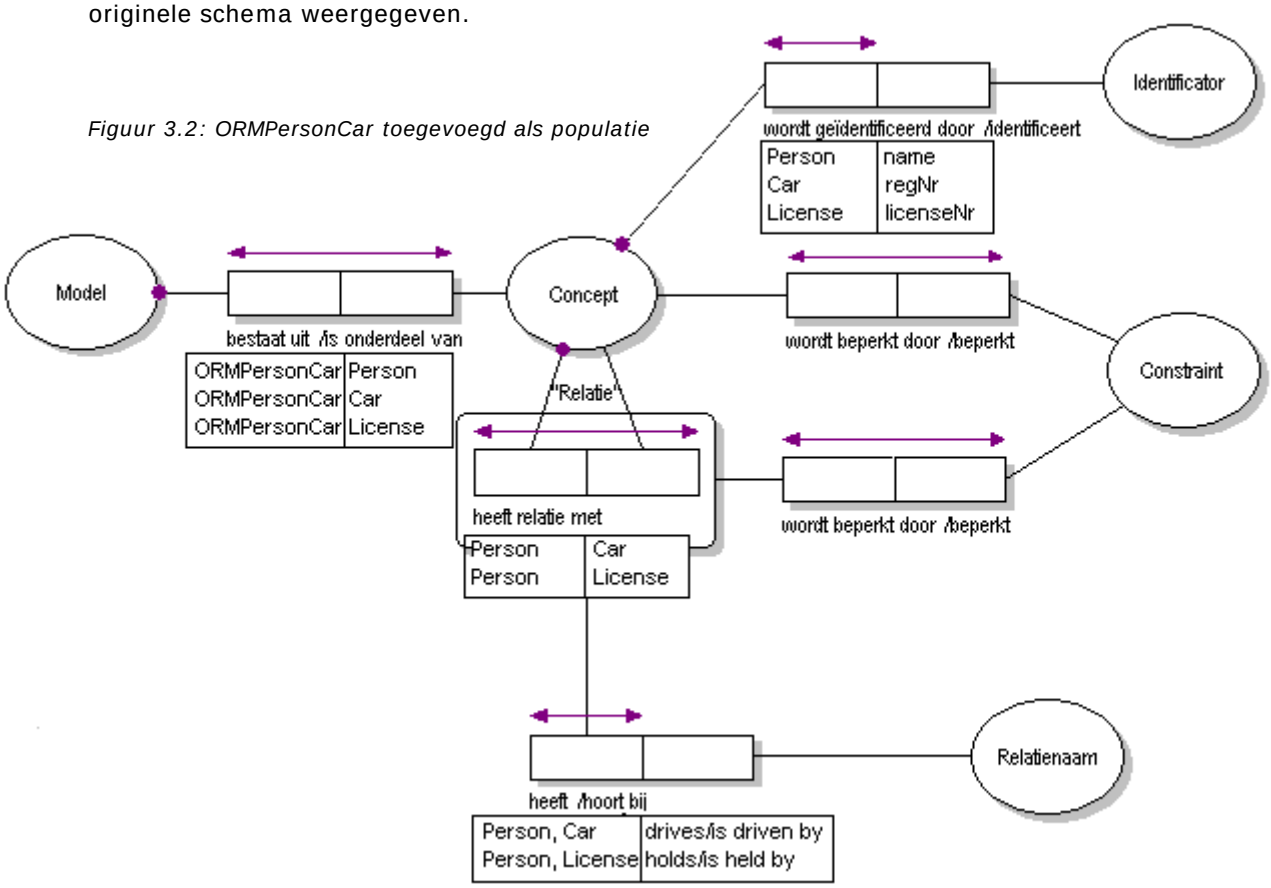
Het eerste model dat als populatie toegevoegd wordt, is onderstaand ORM model:



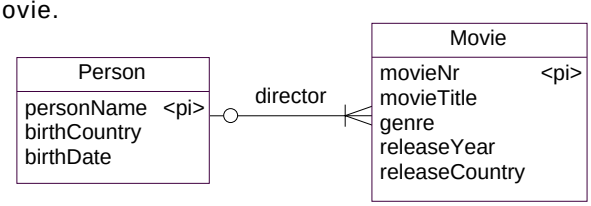
Figuur 3.1: Testpopulatie voor ORM model versie 0.1, ORMPersonCar

Bovenstaand model bestaat uit de concepten “Person”, “Car” en “License”. Deze worden in het schema als populatie van “Concept” toegevoegd. De concepten worden geïdentificeerd door respectievelijk “name”, “regNr” en “licenseNr”. Deze identificatie past nog niet in het schema, vandaar dat er een objecttype “Identificator”wordt toegevoegd, evenals een feittype “Concept wordt geïdentificeerd door/ identificeert Identificator”.
Het toevoegen van de relaties tussen de concepten in het schema levert geen problemen op. Echter, de relaties zijn voorzien van een naam, en die kan niet opgenomen worden in het schema. Daarom wordt er een objecttype “Relatienaam” toegevoegd. Deze wordt aan het concept “Relatie” verbonden door het feittype “Relatie heeft/hoort bij Relatienaam”.
Het resultaat is het volgende gepopuleerde schema. Er wordt slechts een gedeelte van het originele schema weergegeven.

Figuur 3.2: ORMPersonCar toegevoegd als populatie



Het volgende model dat toegevoegd wordt aan de populatie, is een eenvoudig ER diagram, genaamd ERPersonMovie.



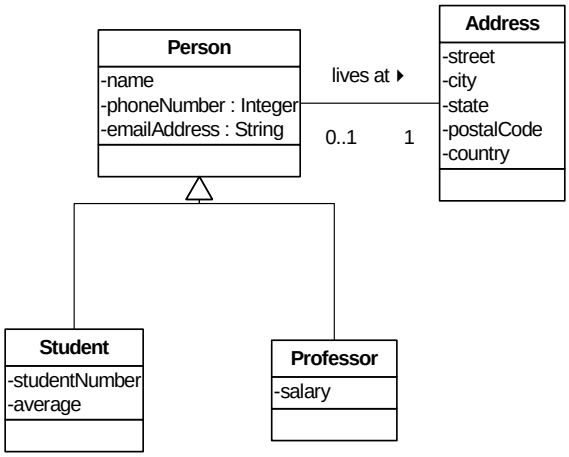
Figuur 3.3: ERPersonMovie

Het toevoegen van de concepten van dit model aan het feittype “Model bestaat uit/is onderdeel van Concept” levert geen problemen op. Echter, het toevoegen van de relatie tussen “Person” en “Movie” levert wel een probleem op. Er bestaat al een relatie tussen “Person” en “Car” en tussen “Person” en “License”. Door het toevoegen van nog een relatie waar “Person” deel van uit maakt, is het niet meer duidelijk welke relatie bij welk model behoort. Bij het opslaan van een relatie zal ook opgeslagen moeten worden bij welk model dit behoort. Om dit te bewerkstelligen, wordt er een rol toegevoegd aan het geobjectificeerde feittype “Relatie”, die verbonden wordt met het objecttype “Model”. De naam van de relatie kan zonder problemen als populatie worden toegevoegd.

In eerste instantie was het idee om een attribuut als een speciaal soort relatie toe te voegen aan het model. Bij nader inzien wordt attribuut toch als een apart objecttype toegevoegd, aangezien er dan ook constraints voor de attributen toegevoegd kunnen worden. Wanneer een feittype tussen “Concept” en “Attribuut” toegevoegd zou worden, zou hetzelfde probleem ontstaan als bij het feittype “Relatie” het geval was, namelijk dat het niet duidelijk zou zijn van welk model deze concepten deel uit zouden maken. Daarom wordt het feittype tussen “Model” en “Concept” geobjectificeerd, en verbonden met het nieuwe feittype. Ook voor “Identificator” geldt dat deze gekoppeld moet worden aan de objectificatie van “Model” en “Concept”, in plaats van aan “Concept”.
Door een uniciteit constraint toe te voegen aan de kant van “Concept in model”, wordt er afgedwongen dat elk concept hooguit één identificator heeft. Door een subset constraint toe te voegen naar “Attribuut is eigenschap van/wordt beschreven door Concept”, is het alleen mogelijk dat de identificator ook daadwerkelijk een attribuut is van “Concept”.

Het laatste dat nog toegevoegd moet worden, zijn de eigenschappen van de relatie. Er moet zowel gekeken worden naar de cardinaliteit van de relatie, als aan welke kant de relatie verplicht is. Het oorspronkelijke objecttype “Constraint” die door een feittype aan “Relatie” gekoppeld was, is hiervoor niet specifiek genoeg, en wordt verwijderd.
Als eerste wordt er een feittype toegevoegd waarmee aangegeven kan worden aan de kant van welk concept de relatie verplicht is. Dit feittype wordt verbonden met zowel “Relatie” als “Concept”. Vervolgens wordt er een objecttype “Cardinaliteit” toegevoegd, dat middels een feittype verbonden wordt aan “Relatie”. Het feittype geeft aan wat de cardinaliteit is van de betreffende relatie, in dit geval is de relatie “ERPersonMovie, Person, Movie” één op veel, wat weergegeven wordt als 1:n. Er wordt vanuit gegaan dat de relatie gelezen wordt zoals deze is ingevoerd bij de relatie, in dit geval dus van “Person” naar “Movie”. Dat is dan ook de manier waarop de cardinaliteit gelezen dient te worden.

Er zal een derde diagram toegevoegd worden als populatie, te weten een UML klassendiagram. Het diagram, UMLPerson, is te vinden op de volgende bladzijde.

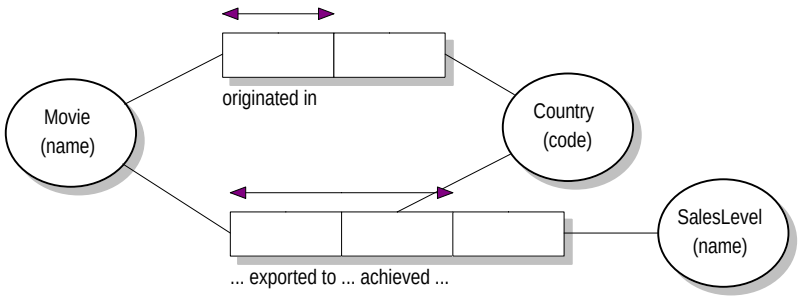


Figuur 3.4: UMLPerson

Allereerst worden de klassen van het UML klassendiagram als populatie van “Concept” toegevoegd. De subtypen van “Person”, te weten “Student” en “Professor”, worden ook als concepten toegevoegd. Vervolgens krijgen de attributen van de concepten een plaats als populatie in het model. Op zich brengt dit geen problemen met zich mee, echter over enkele attributen van “Person” is extra informatie beschikbaar, en die is niet toe te voegen in het schema. Om dit op te lossen wordt er een objecttype “Datatype” toegevoegd. Omdat een attribuut bij meerdere concepten voor kan komen, en bij deze concepten verschillende datatypen kan hebben, wordt het nieuwe objecttype door middel van een feittype verbonden aan de objectificatie van “Attribuut is eigenschap van/wordt beschreven door Concept in model”.

Vervolgens worden de relaties toegevoegd. Er zijn meerdere relaties terug te vinden in UMLPerson, namelijk één ‘normale’ relatie en twee subtype relaties. De normale relatie is het eenvoudigst, dus hier wordt als eerste naar gekeken. De relatie en relatienaam kunnen eenvoudig gepopuleerd worden. Vervolgens wordt gekeken aan welke kant de relatie verplicht is, dat is aan de kant van “Person”; een “Address” hoort altijd bij een “Person”. Aan de kant van “Address” is de relatie niet verplicht, dus hier hoeft niks mee gedaan te worden. De cardinaliteit is 1:1.
De subtypen kunnen gezien worden als een speciaal soort relaties. Om deze echter te kunnen onderscheiden van andere relaties, wordt er een nieuw, tertiair feittype toegevoegd, met de naam “In Model is Concept subtype van Concept”.

Nu dit derde model als populatie toegevoegd is aan het ORM model, zijn drie verschillende soorten modellen gebruikt als populatie. Gezien het feit dat het eerst toegevoegde model, ORMPersonCar erg eenvoudig was, wordt er eerst nog een wat uitgebreider ORM model toegevoegd, voordat er een nieuwe versie van het ORM model weergegeven wordt.
Het vierde toe te voegen model is ORMMovie, dat hieronder weergegeven is.

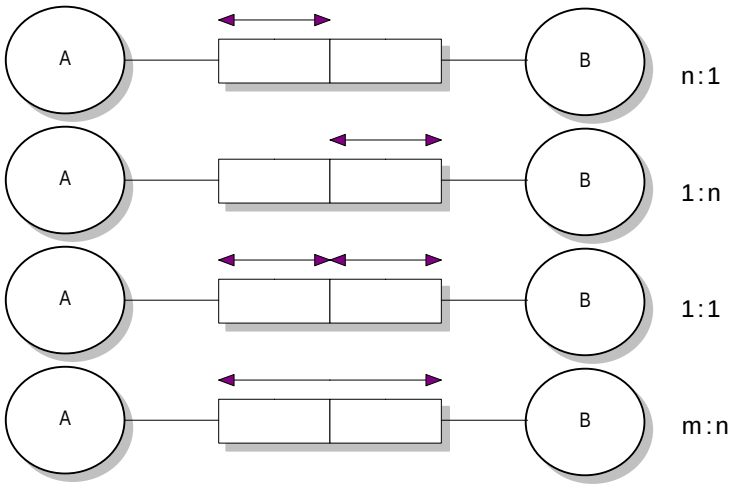


Figuur 3.5: ORMMovie

Het populeren wordt gestart met het toevoegen van de drie concepten in dit model, te weten “Movie”, “Country” en “SalesLevel”. Vervolgens worden respectievelijk “name”, “code” en “name” toegevoegd als attributen van deze concepten. Gezien het feit dat deze attributen de concepten ook identificeren, worden ze ook toegevoegd als identificator van de concepten. Hierna wordt de relatie tussen “Movie” en “Country” toegevoegd, evenals de naam van deze relatie.

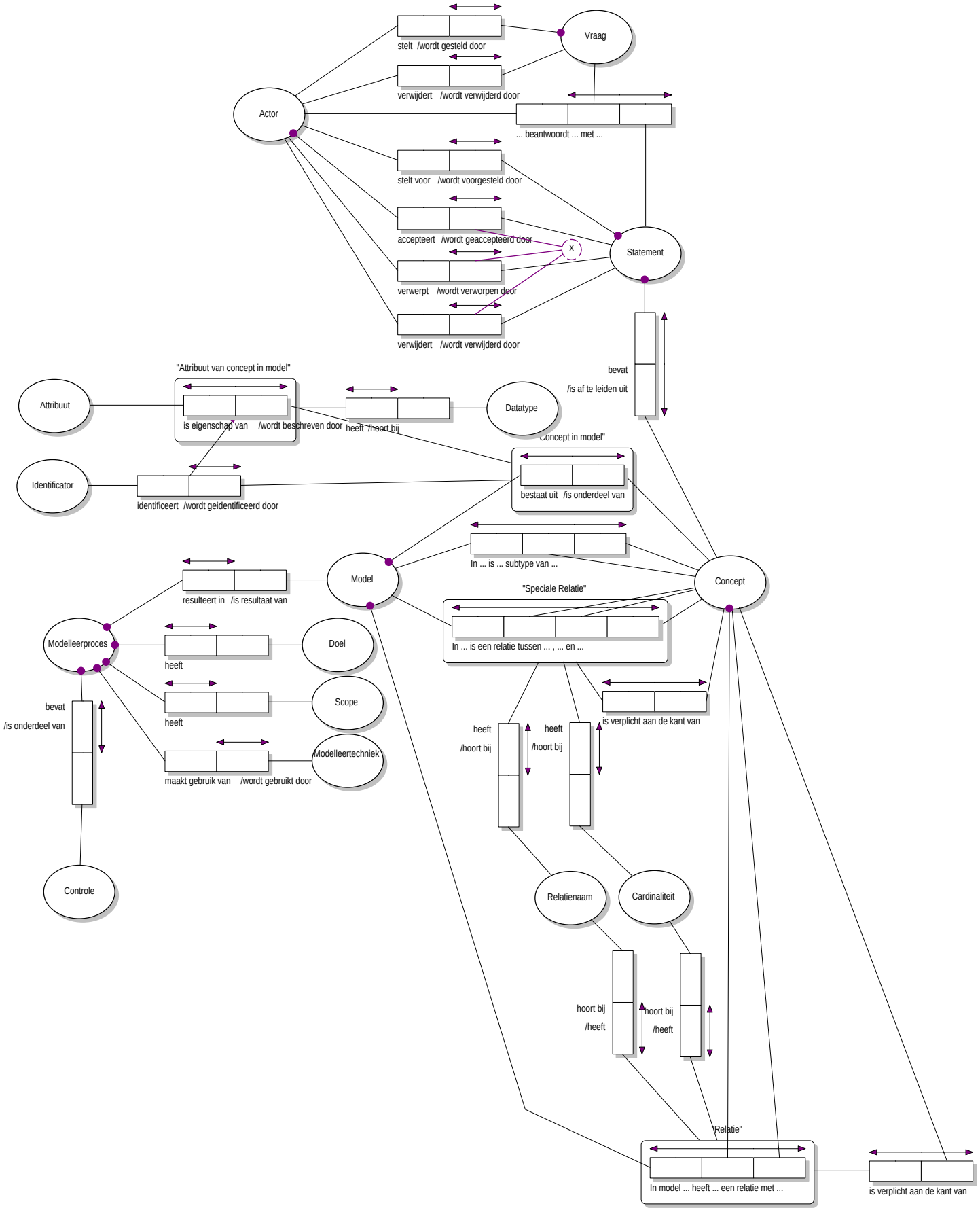
Om de andere relatie toe te kunnen voegen, moet er een wijziging plaatsvinden in het model. Het is namelijk in de huidige versie van het model niet mogelijk om een tertiair feittype toe te voegen, oftewel een relatie tussen drie concepten. Er wordt een viervoudig feittype toegevoegd om dit op te lossen. Dit feittype lijkt op het bestaande feittype “Relatie”, maar heeft een extra rol, om een extra concept op te kunnen slaan in de relatie. Het gevolg hiervan is dat ook van deze speciale relatie een naam vastgelegd moet kunnen worden, evenals de cardinaliteit en eventuele verplichte rollen.
Om dit uit te kunnen voeren wordt het feittype geobjectificeerd en krijgt het de naam “Speciale Relatie”. Vervolgens worden bovengenoemde objeccttypes en relaties toegevoegd. De relatie tussen “Movie”, “Country” en “SalesLevel” kan nu worden toegevoegd. Ook de relatie van de naam kan zonder problemen worden ingevuld.

Het laatste dat moet worden toegevoegd uit dit model zijn de uniciteitconstraints.
De uniciteitconstraints kunnen ingevuld worden bij “Relatie heeft/hoort bij Cardinaliteit”. Bij binaire feittypen zijn er vier manieren om uniciteitconstraints toe te voegen. Deze manieren zijn hieronder weergegeven, met daarnaast de cardinaliteit die bij het desbetreffende voorbeeld hoort [Hal].



Figuur 3.6: Uniciteitconstraints in ORM

Na het toevoegen van dit vierde model, wordt het tijd om een nieuwe versie van het ORM model weer te geven, aangezien er gedurende het populeren van het ORM model met respectievelijk ORMPersonCar, ERPersonMovie, UMLPerson en ORMMovie diverse uitbreidingen en wijzigingen zijn geweest op het ORM model.
ORM model versie 0.2 kunt u vinden op de volgende bladzijde.



Figuur 3.7: ORM model versie 0.2

4. Interviews

Een tweede manier waarop de modellen gevalideerd worden, is door middel van interviews. Van de interviews zal eerst een algemeen verslag weergegeven worden, waarna de inhoud ervan teruggekoppeld zal worden naar de modellen.

4.1 Interview dhr. De Vries

4.1.1 Algemeen verslag

Het eerste interview vond plaats met dhr. De Vries. Dhr. De Vries heeft de opleiding Bedrijfskundige Informatica gevolgd op de HEAO en is daarna doorgegaan met de opleiding Informatie- en Kennistechnologie. Na wat omzwervingen in het bedrijfsleven is hij inmiddels als zelfstandig consultant werkzaam. Zijn eigen bedrijf Zetetic is een adviesbureau in prestatieverbetering door kennismanagement. Hij houdt zich met name bezig met werkzaamheden op het gebied van kennismanagement en kennistechnologie, maar ook op het gebied van informatie en processen. Daarnaast geeft dhr. De Vries een masterclass kennismanagement voor ProEducation, een onderdeel van de Hogeschool van Amsterdam.

Binnen zijn werkzaamheden maakt dhr. De Vries regelmatig gebruik van modellen. Bijvoorbeeld bij een soort brainstormachtige sessies, waar uiteindelijk een soort mindmap uit ontstaat. Ook op het gebied van processen, informatiestromen en informatiesystemen worden modellen gebruikt. De techniek die hiervoor gebruikt wordt, is niet alleen afhankelijk van het doel van het model, maar ook van wat er gebruikelijk is in de organisatie. Wanneer de organisatie van een bepaalde modelleertechniek gebruik maakt, past dhr. De Vries zich aan deze techniek aan. Vaak zijn dit vooral standaardtechnieken als processtroomdiagrammen, gegevensmodellen en UML. Voor kennismodellering geldt dit in mindere mate, omdat hier nog weinig standaardmethoden voor zijn.

Binnen het gebruik van de verschillende technieken, is er een bepaalde lijn te herkennen, een bepaalde manier van werken, die dhr. De Vries prettig vindt, en waarvan hij weet dat deze in bepaalde situaties goed werkt. Allereerst probeert hij duidelijk te krijgen waarvoor hij ingehuurd is, wat het probleem is, en dus ook wat het doel van het model is. Zijn voorkeur gaat ernaar uit om dit vanuit één iemand duidelijk te krijgen, namelijk de opdrachtgever. Het duidelijk krijgen wie de opdrachtgever is, is daarom erg belangrijk. Aan de hand van het doel van het model, en met behulp van de reeds in de organisatie gebruikte modelleertechnieken wordt een modelleertechniek gekozen. Daarnaast is het ook belangrijk om te weten welke uitgangspunten leidend zijn bij het maken van het model, deze dienen daarom in een vroeg stadium bepaald te worden.

Dhr. De Vries vindt het prettig om vervolgens een workshop te houden met bepaalde mensen, die te maken hebben met een proces, systeem of bepaald domein. In het begin zijn deze sessies op een vrij hoog niveau om erachter te komen wat nou de zaken zijn die van belang zijn. Deze zaken worden aangestipt en, eventueel in een volgende sessie, uitgediept. In sommige organisaties werkt men liever met interviews, waarbij individuele gesprekken plaatsvinden. Dit kost meer tijd, en het is moeilijker om iedereen met de neus dezelfde kant op te krijgen. Tijdens zo'n sessie is het belangrijk dat de belangrijkste stakeholders bij elkaar komen. Daarom dient er eerst geïdentificeerd te worden wie die stakeholders zijn. Dit wordt in samenspraak met de opdrachtgever bepaald. Een modelleersessie is zinvol wanneer er maximaal tien tot twaalf mensen aanwezig zijn, het ideale aantal is over het algemeen een stuk of acht.

De aanpak die dhr. De Vries gebruikt binnen een workshop is afhankelijk van in hoeverre reeds helder is wat de opdracht is. Een tactiek die hij gebruikt op het moment dat het hem nog niet helemaal helder is, is een yellow paper sessie; de aanwezigen krijgen allemaal een aantal gele papiertjes, waar ze op kunnen schrijven welke dingen ze bij een bepaald onderwerp belangrijk vinden, of welke problemen ze bij het onderwerp ervaren. Door deze briefjes op te hangen, ontstaat er redelijk snel een gezamenlijk beeld van het onderwerp of probleem.

Deze werkwijze wordt met name in een eerste sessie gehanteerd, om een beeld te krijgen van het probleem. In een tweede sessie worden de dingen uitgediept, waarna de achterhaalde informatie geverifieerd en afgestemd wordt. Hierbij gaat dhr. De Vries nog regelmatig naar bepaalde mensen toe om dingen af te stemmen of te verhelderen. Vaak komen bij het uitwerken nog dingen naar boven die of voor hemzelf, of zelfs voor de gehele organisatie nog niet duidelijk zijn. Er kan dan een issuelijst ontstaan, met punten die of in een modelleersessie of met de opdrachtgever nog besproken moeten worden.

Wanneer de juiste mensen aanwezig zijn bij een modelleersessie, dan ontstaat er over het algemeen weinig discussie over hoe bepaalde problemen opgelost moeten worden. Elke aanwezige heeft zijn eigen expertise, en die expertise wordt door de rest van de aanwezigen onderkend. Een domeindeskundige bepaalt dan wat er wel en niet kan in zijn domein en vervolgens bepalen de rest van de deskundigen wat voor impact dat heeft op hun domein. Tevens wordt er gekeken of er nog aan de basisprincipes wordt voldaan. Wanneer er discussies ontstaan, betreft dit vaak raakvlakken tussen verschillende systemen. Wanneer er tijdens een workshop een issue boven tafel komt waar geen eenduidige oplossing voor gevonden wordt door de aanwezigen, dan worden er diverse scenario's bedacht met hun eigen voor- en nadelen en vervolgens wordt de issue tijdelijk geparkeerd. Het is dan aan de opdrachtgever om hier een beslissing over te nemen. Wanneer het echter om kleine zaken gaat, kan de domeindeskundige hier een beslissing over nemen.

Tijdens een modelleersessie wordt er vooral veel opgeschreven. Daarnaast wordt er een foto gemaakt van de yellow papers. Ook door middel van mindmaps wordt veel informatie vastgelegd. Dit alles wordt gebruikt als achtergrondinformatie en communicatiemiddel naar de organisatie. Naast het opstellen van het model is het ook belangrijk om een beschrijving te maken in natuurlijke taal. Bij een proces wordt bijvoorbeeld beschreven wat de input is van elke stap, het doel van die stap, de systemen die daarbij geraakt worden, het resultaat van die stap etcetera.

Het controleren van het model wordt over het algemeen uitgevoerd door het opnieuw goed doorlopen van alle dingen. De domeindeskundigen kijken er naar, spreken het door en leveren hun commentaar erop. Daarnaast zijn er per techniek wel een paar vuistregels om te controleren of het model klopt. Als bij een processtroom diagram een lijntje los hangt, of een proces geen output heeft, dan klopt er waarschijnlijk iets niet.
Van elk nieuw model en elk nieuw document wordt een nieuwe versie opgeslagen. Daarnaast wordt elke wijziging beschreven. Wanneer het een omvangrijk document bevat, worden de wijzigingen op een wijzigingsblad genoteerd, omdat de wijzigingen anders moeilijk terug te vinden zijn.

Een definitief model bestaat niet, het kan altijd nog veranderen, ook al is dat soms pas twee jaar later, als er bijvoorbeeld een nieuw proces geïntroduceerd wordt. Een model is alleen definitief voor een bepaald moment. Wanneer alle onduidelijkheden opgelost zijn, en verwerkt zijn in het model, en iedereen die het model moet gaan gebruiken, vindt het model bruikbaar, dan is het model klaar voor dat moment. Over het algemeen is dat na drie à vier iteratieslagen.

Na afloop van het project bewaart dhr. De Vries alle informatie die gedurende het project verzameld is. Of de opdrachtgever dit ook bewaart, is afhankelijk van de organisatie. Sommige organisaties springen hier heel nauwkeurig mee om en hebben van alle projecten een projectdossier. Andere organisaties zijn hier niet zo netjes in.
Dhr. De Vries gebruikt deze informatie nog voor andere projecten. Hij heeft een soort bibliotheek van methoden, waarvan hij weet dat deze in een bepaalde situatie goed werken. Met name bij kennismodelleren is dit erg bruikbaar.

4.1.2. Koppeling naar de modellen

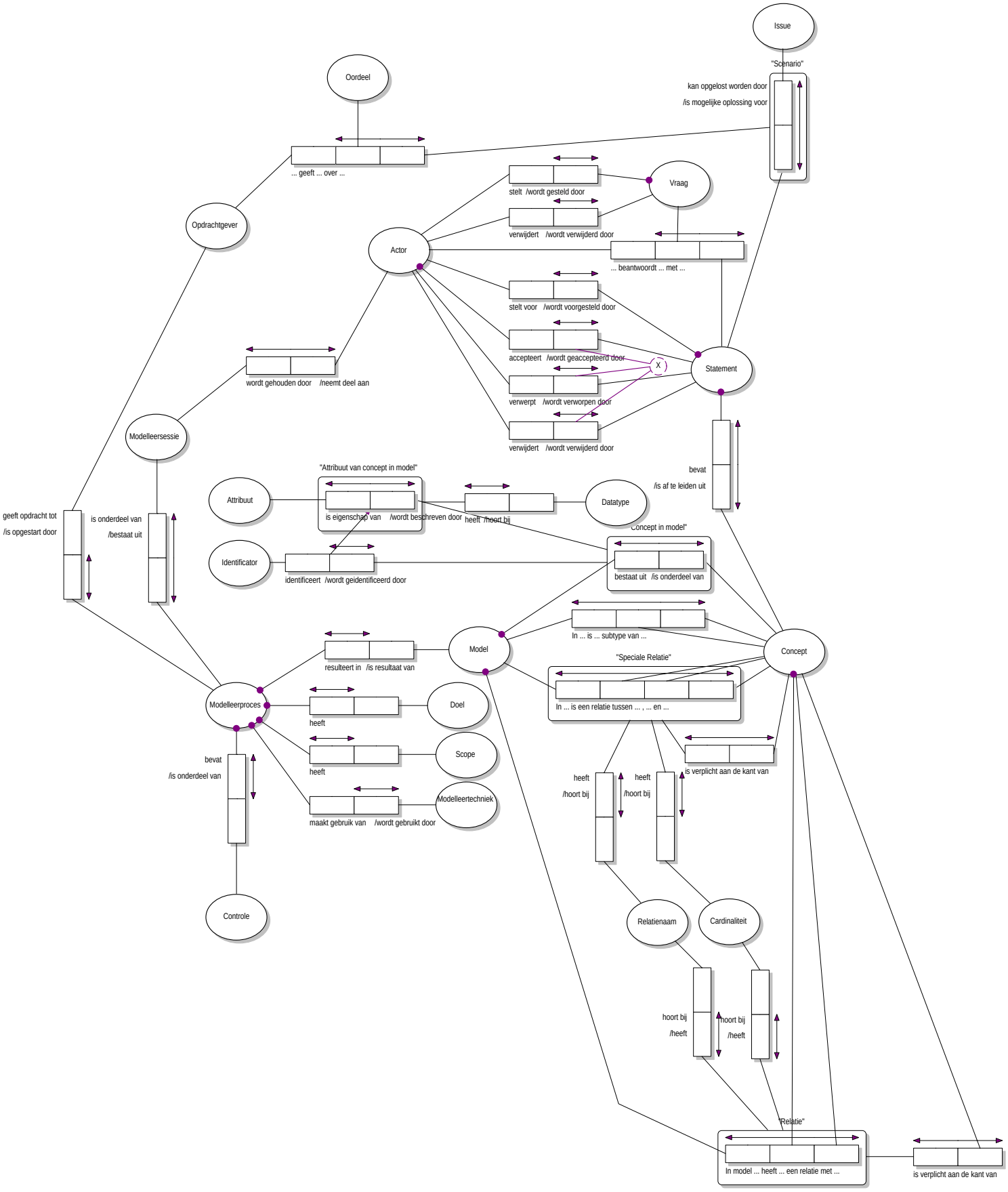
Het modelleerproces dat dhr. De Vries volgt, zoals hierboven beschreven, komt in grote lijnen overeen met het in paragraaf 2.4 beschreven proces. Er zijn zowel enkele nuanceverschillen als kleine verschillen die zullen leiden tot enkele aanpassingen in de modellen.

Het identificeren van de stakeholders is een belangrijk punt, omdat door het identificeren van de juiste stakeholders een modelleersessie beter zal verlopen. Er zal dan minder discussie ontstaan. In het oorspronkelijk beschreven modelleerproces is het identificeren van de stakeholders opgenomen in de domeinscan. Dit levert geen veranderingen in de modellen op. In het ORM model is een stakeholder terug te vinden als actor in een modelleersessie. Een actor kan elke aanwezige zijn in een modelleersessie, zowel een domeinexpert als een systeemanalist. Hoewel met stakeholder niet de systeemanalist bedoeld wordt, is het niet nodig om actor op te splitsen in stakeholder en systeemanalist, omdat naast hun naam geen extra informatie over de stakeholders vastgelegd wordt.

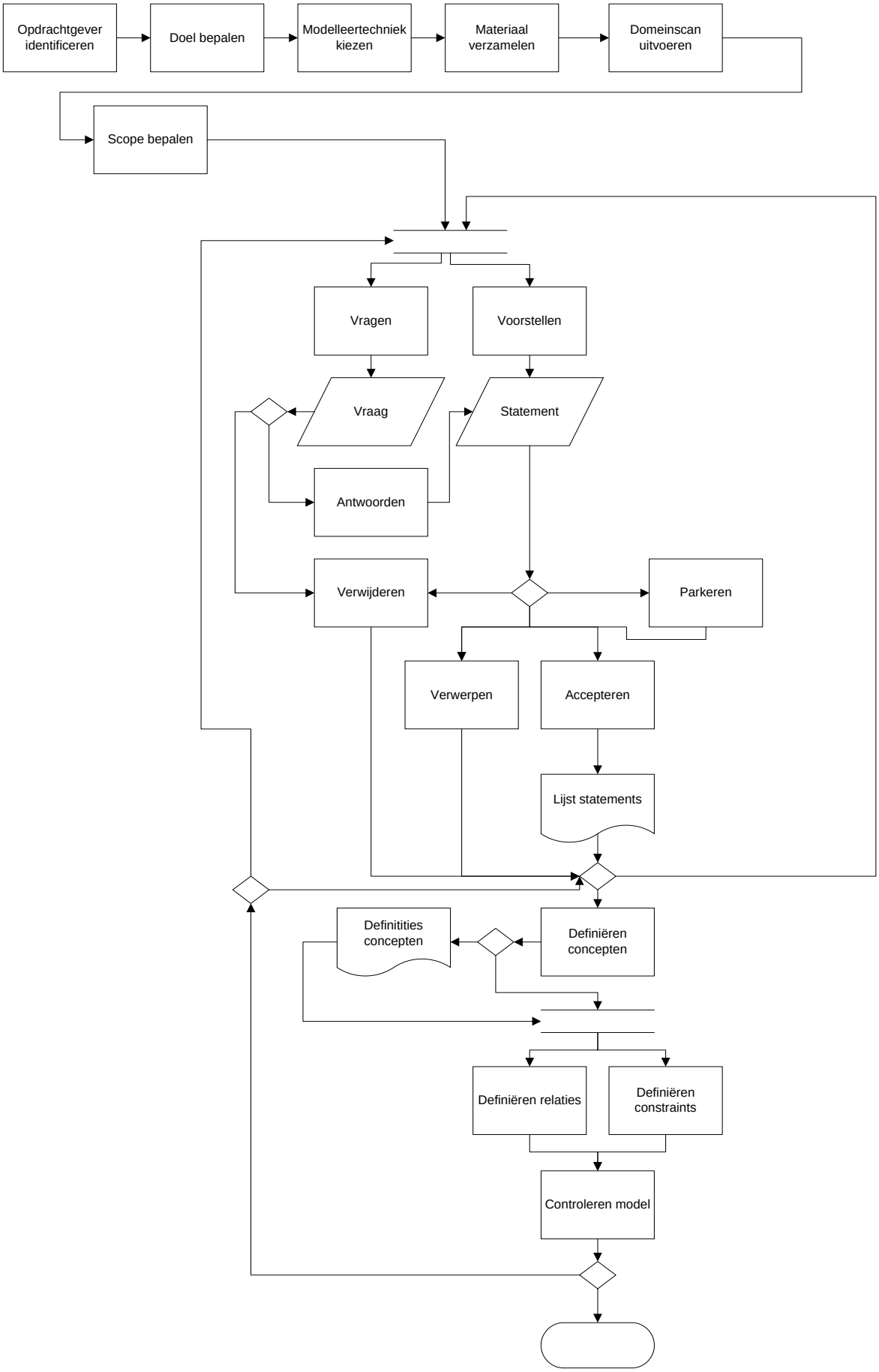
Een punt dat wel door dhr. De Vries wordt beschreven, maar dat momenteel nog niet in het model terug te vinden is, is het identificeren van de opdrachtgever. Deze komt in het proces naar voren wanneer er beslissingen moeten worden genomen over bepaalde issues, die niet in de modelleersessie genomen kunnen worden. Dit is een belangrijk onderdeel van het modelleerproces, en zal aan beide modellen toegevoegd worden.
Om het onderscheid te kunnen maken tussen het nemen van beslissingen binnen de modelleersessie en daarbuiten, zal er in het ORM model een objecttype “Modelleersessie” toegevoegd worden, die door middel van twee feittypen verbonden wordt met zowel “Modelleerproces” als “Actor”. Zo kunnen er meerdere modelleersessies onderscheiden worden binnen een modelleerproces en kan ook vastgelegd worden welke actoren in een modelleersessie aanwezig zijn.

Wanneer er een issue ontstaat die binnen een modelleersessie niet opgelost kan worden, dan worden er verschillende scenario’s bedacht, waar uiteindelijk een beslissing over wordt genomen door de opdrachtgever. Dit is in de huidige modellen nog niet terug te vinden en zal dus worden toegevoegd. In het ORM model betekent dit dat er een objecttype “Issue” wordt toegevoegd, die door een feittype wordt verbonden met “Modelleersessie”. Voor een issue kunnen meerdere scenario’s worden bedacht. Deze scenario’s kunnen gezien worden als statements, die voorgesteld worden. In een feittype tussen “Issue” en “Statement” kan vastgelegd worden welk statement bij welke issue hoort. Dit feittype wordt geobjectificeerd en verbonden met “Opdrachtgever”. Die kan op die manier over alle voorstellen een besluit nemen.

De nieuwe versies van de modellen zijn te vinden op de volgende bladzijden.



Figuur 4.1: ORM model versie 0.3



Figuur 4.2: Flowchart versie 0.2

4.2 Reflectie tussen de interviews door

Gedurende de periode van interviews, is er tussen de interviews door ruimte voor een eigen reflectie op de modellen. Deze reflectie is op te delen in twee subreflecties, te weten een reflectie van het modelleerproces, en een reflectie van het gedeelte waarin modellen worden opgeslagen.

4.2.1 Reflectie op het modelleerproces

Om het gedeelte van het ORM model te controleren waarin het modelleerproces wordt opgeslagen, inclusief de modelleersessie, worden er twee voorbeeld modelleerprocessen bedacht, en als populatie toegevoegd aan het model.

4.2.1.1 Voorbeeld modelleerproces 1

Het eerste modelleerproces is genaamd “In kaart brengen van bedrijf X”. De opdrachtgever van dit proces is de directeur van bedrijf X, dhr. A. Janssen. Het doel van het proces is het in kaart brengen van de structuur van het bedrijf. De huidige structuur kan dan vervolgens geanalyseerd en verbeterd worden om de problemen die nu voorkomen in het bedrijf op te lossen. Binnen het bedrijf zijn de afdelingen niet hetzelfde als de bedrijfsprocessen; het komt voor dat binnen een afdeling zich meerdere processen afspelen, en dat een bedrijfsproces over meerdere afdelingen verspreid is. De scope betreft slechts de structuur en hiërarchie van de organisatie, de bedrijfsprocessen vallen er buiten. De modelleertechniek die hier voor gebruikt zal gaan worden is een UML klassendiagram.
Om tot een model te komen worden er modelleersessies gehouden. Hieronder is een deel van de eerste modelleersessie terug te vinden.

Voorstellen	DE	Janssen werkt voor verkoop.
Vragen	SA	Wat voor ding is verkoop?
Voorstellen	DE	Verkoop is een afdeling.
Vragen/Voorstellen	SA	Onderscheiden we verkoop van andere afdelingen door middel van zijn naam?
Accepteren	DE, SA	Ja [we onderscheiden verkoop van andere afdelingen door middel van zijn naam.]
Vragen/Voorstellen	SA	Bent u het er mee eens dat verkoop een afdeling is met naam verkoop?
Accepteren	DE, SA	Ja [verkoop is een afdeling met naam verkoop.]

Figuur 4.3: Modelleersessie bij modelleerproces “In kaart brengen van bedrijf X”

Dit deel van de modelleersessie wordt toegevoegd als populatie aan ORM model versie 0.3. Hetzelfde geldt voor de naam van het modelleerproces, opdrachtgever, scope, modelleertechniek en doel. Het resulterende model zal niet toegevoegd worden, omdat het hier gaat om het controleren van het ORM model op het gebied van het modelleerproces zelf.

De modelleersessie en alle bijbehorende zaken kunnen zonder problemen toegevoegd worden. De twee regels in het voorbeeld waarbij de systeemanalist in één keer zowel een vraag stelt als een voorstel doet, worden in het model eerst als vraag toegevoegd, waarna de systeemanalist zelf antwoord geeft op de vraag door middel van het statement dat in de latere regels geaccepteerd wordt. Om dit duidelijk te maken wordt de populatie van de feittypen van de eerste “vragen/voorstellen” regel hieronder weergegeven:

Bij “Actor stelt/wordt gesteld door Vraag” wordt de volgende populatie toegevoegd:

SA	*Onderscheiden we verkoop van andere afdelingen door middel van zijn naam?*

Vervolgens komt bij “Actor beantwoordt Vraag met Statement” de volgende populatie:

SA	*Onderscheiden we verkoop van andere afdelingen door middel van zijn naam?*	*We onderscheiden verkoop van andere afdelingen door middel van zijn naam.*

In de huidige versie van het model staat een verplichte rol constraint op “Statement” bij “Actor stelt voor/wordt voorgesteld door Statement”. Dit voorbeeld maakt echter duidelijk dat een statement niet altijd direct voorgesteld wordt, maar dat dit ook kan gebeuren door middel van een antwoord op een vraag. In dat geval wordt het statement niet toegevoegd als populatie bij “Actor stelt voor/wordt voorgesteld door Statement”. De verplichte rol constraint klopt dus niet, en moet aan de kant van “Statement” gezet worden op de combinatie van “Actor stelt voor/wordt voorgesteld door Statement” en “Actor beantwoordt Vraag met Statement”, omdat dit de twee mogelijkheden zijn om een statement te introduceren.

Het toevoegen van de populatie levert op de wijziging van de verplichte rol constraint geen veranderingen op. Het toevoegen van de populatie maakt echter wel duidelijk dat wanneer er meerdere modelleerprocessen worden opgeslagen, er wel problemen zullen komen. Bij modelleersessie 1 van bovenstaand modelleerproces, werden de aanwezigen SA (systeemanalist) en DE (domeinexpert) opgeslagen. Bij een ander modelleerproces kan echter ook een modelleersessie 1 voorkomen, en ook de aanwezigen kunnen overeenkomen. Het is dan niet meer te onderscheiden wie bij welke modelleersessie van welk modelleerproces aanwezig was. Hetzelfde geldt voor het voorstellen, accepteren, verwijderen en verwerpen van statements en het stellen en beantwoorden van vragen. Bij het toevoegen van het tweede modelleerproces zal dit aangepast worden.

4.2.1.2 Voorbeeld modelleerproces 2

Het tweede modelleerproces heeft als naam Pers024. Het doel van dit proces is om eisen voor een nieuw personeelssysteem te modelleren. Het model zal gemaakt worden met behulp van ORM. Het gaat bij het nieuwe systeem alleen om personen die in Nijmegen wonen, personen die buiten Nijmegen wonen, vallen buiten de scope. De opdrachtgever heet mevr. L. de Jong. Er wordt een modelleersessie gehouden waarbij drie personen aanwezig zijn, hieronder aangegeven als SA, DE1 en DE2. De namen van deze personen zijn in dit geval niet belangrijk, hoewel deze bij een echte modelleersessie wèl gebruikt zullen worden in plaats van benamingen als SA en DE.
De modelleersessie is terug te vinden op de volgende pagina.

Voorstellen	DE1	John woont in Nijmegen.
Vragen	SA	Wat voor ding is John?
Voorstellen	DE1	John is een persoon.
Vragen/ Voorstellen	SA	Onderscheiden we John van andere personen door middel van zijn achternaam?
Accepteren	DE1	Ja, we onderscheiden John van andere personen door middel van zijn achternaam.
Verwerpen/ Voorstellen	DE2	Nee, we onderscheiden John niet van andere personen door middel van zijn achternaam, we onderscheiden hem door middel van zijn voornaam.
Issue	SA	Hoe onderscheiden we John van andere personen?
Scenario 1	DE1	We onderscheiden John van andere personen door middel van zijn achternaam.
Scenario 2	DE2	We onderscheiden John van andere personen door middel van zijn voornaam.

Figuur 4.4: Modelleersessie bij modelleerproces "Pers024"

Het toevoegen van het doel, de scope, de modelleertechniek en de opdrachtgever aan het ORM model lukt zonder problemen. Het toevoegen van de modelleersessie bij het modelleerproces gaat ook goed, maar laat wel duidelijk zien dat een modelleersessie niet uniek is. Dit blijkt te kloppen bij het populeren van de aanwezigen bij de modelleersessie. Er zijn twee modelleersessies '1', en bij beide sessies is SA (de systeemanalist) aanwezig. Het is nu echter niet duidelijk bij welke sessie DE, DE1 en DE2 horen. Daarom wordt het feittype in plaats van met "Modelleersessie" nu verbonden met de objectificatie van "Modelleersessie is onderdeel van/bestaat uit Modeleerproces". Hierdoor wordt het duidelijk welke modelleersessie bij welk modelleerproces hoort. Omdat "Actor" niet uniek is, maar slechts uniek is als ook bekend is in de combinatie van modelleerproces en modelleersessie, dienen de feittypen die aan "Actor" hangen, gekoppeld te worden aan de objectificatie van "Modelleersessie van modelleerproces wordt gehouden door/neemt deel aan Actor". Het eerste deel van de modelleersessie kan nu zonder problemen ingevuld worden in het model.

Wanneer er geen uitkomst gevonden kan worden over een bepaald statement, in dit geval hoe John onderscheiden wordt van andere personen, dan ontstaat er een issue. In dit voorbeeld zijn er twee scenario's, namelijk:

- We onderscheiden John van andere personen door middel van zijn achternaam;
- We onderscheiden John van andere personen door middel van zijn voornaam.

Deze issue wordt geparkeerd; er wordt in de modelleersessie geen antwoord op gevonden. De opdrachtgever zal per scenario moeten besluiten of deze de juiste is. Bij het toevoegen van het voorbeeld aan het model, is direct te zien dat hier niet duidelijk is bij welk modelleerproces de issue hoort en in welke modelleersessie deze naar voren is gekomen. Om af te kunnen leiden welke issue bij welke modelleersessie van een modelleerproces naar voren is gekomen, wordt er een feittype gemaakt “Issue komt naar voren in/levert op Modelleersessie van modelleerproces”. Dit feittype wordt geobjectificeerd en vervolgens door middel van het feittype “Issue kan opgelost worden door/is mogelijke oplossing voor Statement” verbonden aan “Statement”. Hier worden de scenario’s in opgeslagen. Een soortgelijk feittype, eveneens tussen “Issue” en “Statement” wordt aangemaakt om daar de uiteindelijke oplossing in op te slaan. De populatie van dit feittype is een subset van de populatie van “Issue kan opgelost worden door/is mogelijke oplossing voor Statement”. Omdat de oplossing van een issue nu al opgeslagen wordt, kan het drievoudige feittype “Opdrachtgever geeft Oordeel over Scenario” vervangen worden door “Opdrachtgever neemt besluit over Issue in modelleersessie van modelleerproces”.
Het tweede voorbeeld is nu compleet verwerkt in het model.

4.2.2 Reflectie op het opslaan van modellen

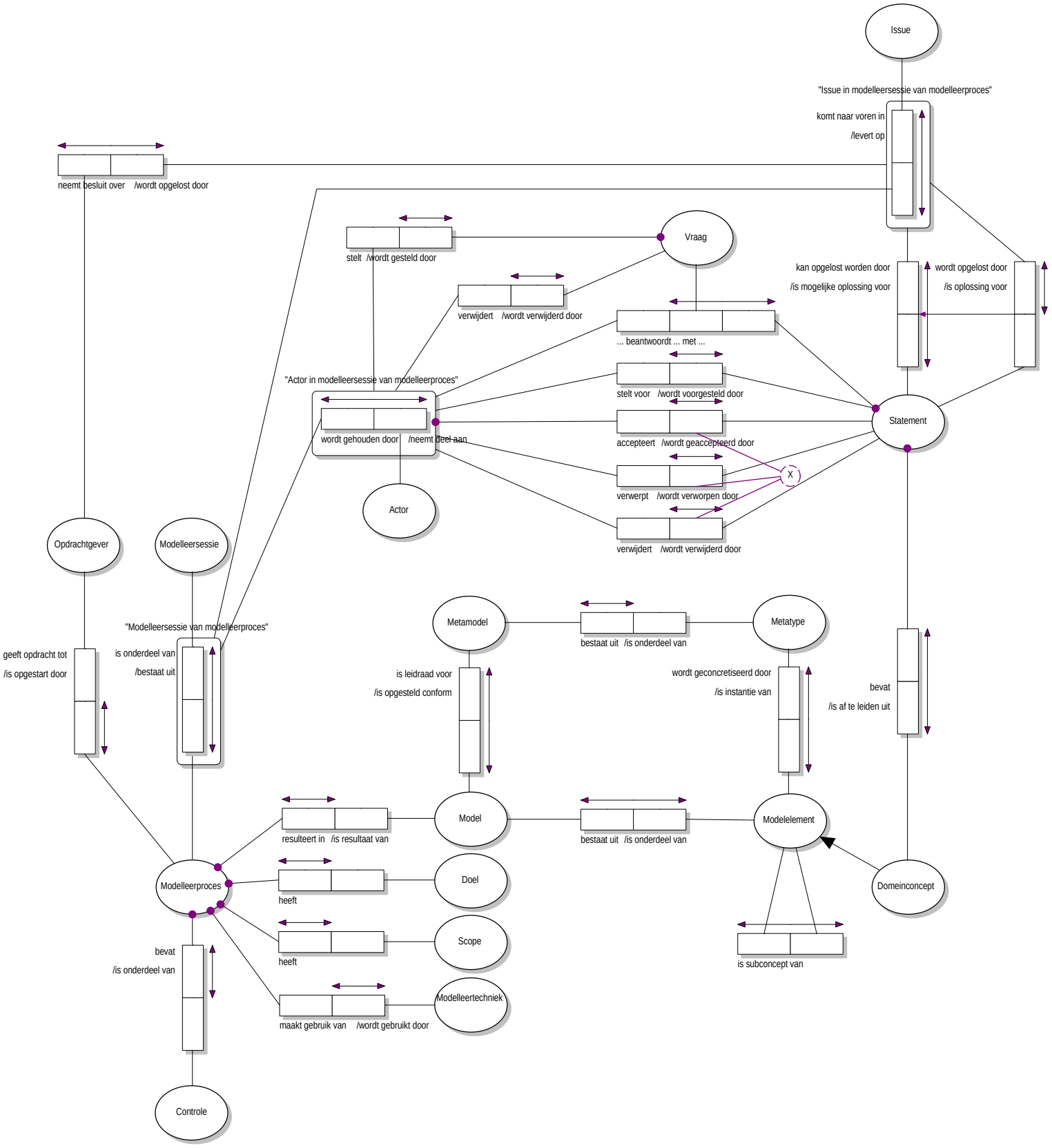
Naast het modelleerproces verdient ook het gedeelte in het ORM model waar het model opgeslagen wordt, enige reflectie. Hoewel er in deze scriptie reeds aandacht is besteed aan het valideren van dit gedeelte van het model, is dit gedeelte niet perfect. Er zijn nog steeds delen van modellen die niet in dit ORM model opgeslagen kunnen worden, zoals objectificatie bij ORM en verschillende soorten constraints bij alle technieken. Dit gedeelte zou nog verder uitgebreid kunnen worden, maar is vanwege de complexiteit in feite niet perfect te krijgen. Daarnaast gaat het in deze scriptie vooral om het modelleerproces, en is dat gedeelte meer bijzaak. Wanneer ook het opslaan van modellen verder onderzocht zou worden, zouden er twee studies naast elkaar gaan lopen.
Het gedeelte in het ORM model waarin de modellen opgeslagen worden, zal om deze reden verwijderd worden, en worden vervangen door een ander, kleiner gedeelte, dat betrekking heeft op metamodellen.

Elke modelleertechniek kent een metamodel. Een metamodel kan gezien worden als een formeel systeem. Een dergelijk systeem bestaat uit twee onderdelen, te weten een uniek kenmerk dat de concepten specificeert, een basis biedend voor de definitie van goed gevormde formules, en een set van zulke goed gevormde formules, ook wel axioma’s genoemd, die aangenomen of verplicht dienen te zijn om te gelden voor concrete systemen die het formele systeem realiseren [PW05].

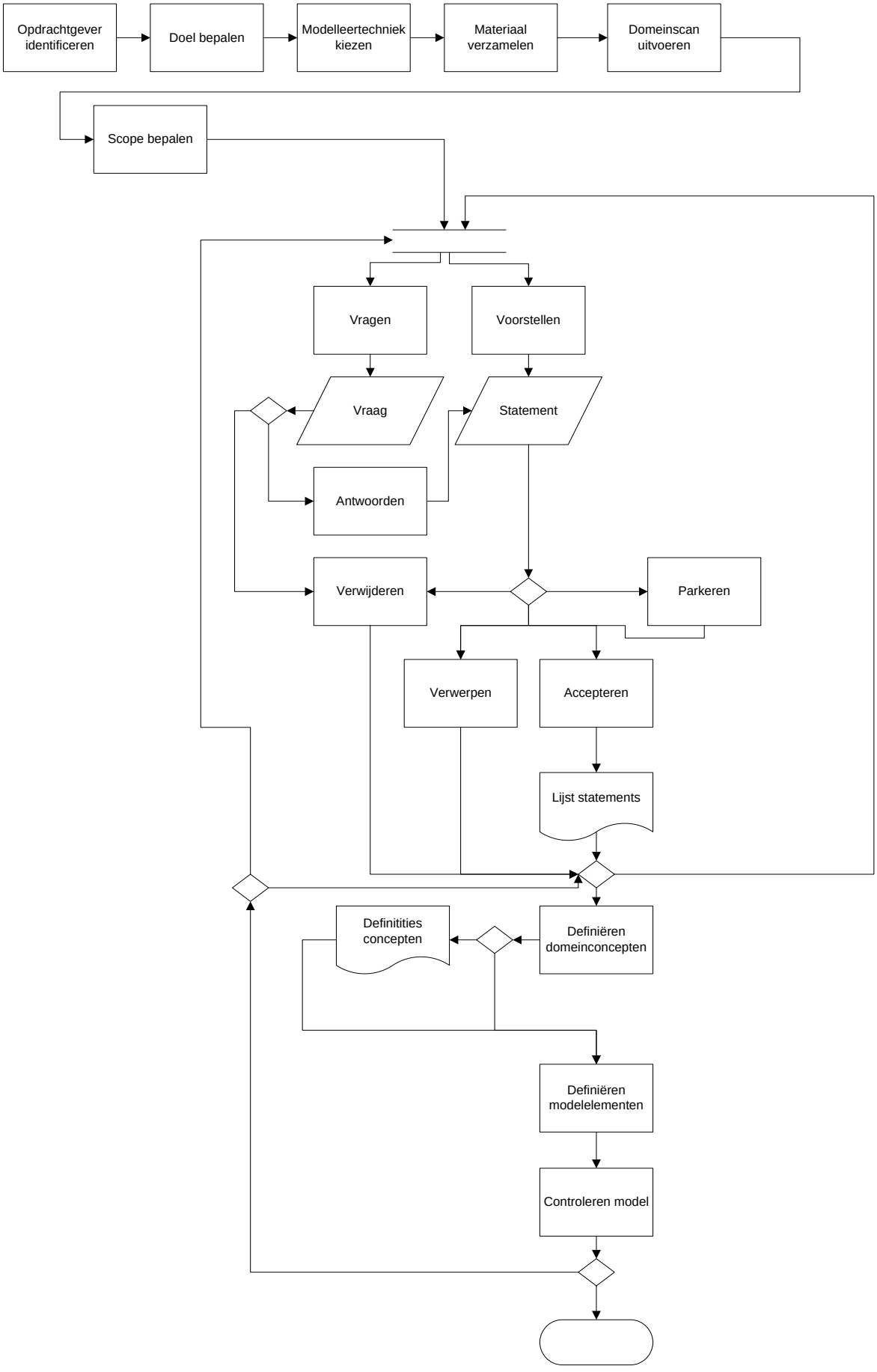
Naar de concepten van het formele systeem wordt gerefereerd als de typen van het metamodel. Een model is een instantie van een formeel systeem, het geassocieerde metamodel. Dit model bevat instanties van de metatypen die opgenomen zijn in het metamodel. Deze set van al deze instanties zijn de model elementen. De mogelijke interpretatie van deze elementen is de combinatie van een model element en een metatype. Omdat metamodellen subtypen kunnen bevatten, kunnen elementen geassocieerd worden met meerdere metatypen [PW05].

Het begrip “Concept” zoals dat in de opgestelde modellen gebruikt wordt, zijn instanties van model elementen zoals die hierboven beschreven zijn. In het ORM model zal dit dus een subtype worden van “Modelelement”. Om verwarring te voorkomen wordt de naam van “Concept” veranderd in “Domeinconcept”, omdat het gaat om een concept zoals dit in het domein voorkomt, en dit geen concept in een metamodel betreft.

Dit leidt tot aanpassingen in zowel het ORM model als de flowchart. De nieuwe versies van beide modellen kunt u vinden op de volgende bladzijden.



Figuur 4.5: ORM model versie 0.4



Figuur 4.6: Flowchart versie 0.3

4.3 Interview mevr. Bleeker

4.3.1 Algemeen verslag

Het tweede interview dat werd gehouden was het interview met mevr. Bleeker. In 1995 voltooide mevr. Bleeker haar studie Informatica. Hierna werkte ze bij verschillende ICT-bedrijven en werkte ze zich op tot informatiearchitect [KPH]. Haar werkzaamheden liggen op het snijvlak tussen business en IT. Ze werkt in commerciële projecten, waarbij ze het liefst zo ver mogelijk aanhaakt aan de businesskant, bijvoorbeeld bij de marketingafdeling of een businessarchitect, om te kijken wat voor visie zij hebben of welke kant zij op willen. Daar sluit mevr. Bleeker bij aan door te kijken wat zij aan IT-ondersteuning nodig hebben. Ze houdt zich hierbij zo min mogelijk bezig met de technische dingen. Het liefst werkt ze samen met een technisch architect die zorg draagt voor alles wat met techniek en realisatie te maken heeft.

Mevr. Bleeker gebruikt bij haar werkzaamheden modellen om duidelijk te krijgen welke informatie of businessconcepten een rol spelen in een domein, daarbij kijkend op welk deel van de informatievoorziening het beste gericht kan worden, en waar de vrijheidsgraden of juist beperkingen zitten. Dit kan zowel gebruikt worden wanneer er een systeem vervangen moet worden, maar ook wanneer het om iets innovatiefs gaat.

Bij het maken van modellen gebruikt mevr. Bleeker ORM als denkwijze. Daarnaast documenteert ze regelmatig in UML, omdat de techneuten daar de voorkeur aan geven. Ook tekstuele beschrijvingen spelen een belangrijke rol, omdat de businessmensen de UML en ORM platen vaak niet kunnen lezen. Bij het opstellen van de modellen komt ervaring goed van pas. Als een bepaald domein bekend terrein is, is het vaak al duidelijk wat voor dingen daar spelen, en dan kan daar direct naar gevraagd worden. Ook interviewtechnieken gebruikt mevr. Bleeker veel om tot een goed model te komen, bijvoorbeeld doorvragen en checkvragen stellen. Vaak heeft ze al snel een gevoel hoe het domein in elkaar zit, en door dit te combineren met interviewtechnieken, worden dingen uitgesloten en kunnen grenzen afgebakend worden. Voorbeelden zijn een goed hulpmiddel om inzichtelijk te maken wat bijvoorbeeld wel en niet kan als voor een bepaalde oplossing wordt gekozen.
Mevr. Bleeker werkt over het algemeen top-down, eerst probeert ze de buitengrenzen vast te stellen van het gedeelte waar het om gaat, vervolgens identificeert ze de subgebieden en die gebieden kunnen dan later in detail uitgewerkt worden.

Als ze inzicht wil krijgen in een domein, geeft mevr. Bleeker de voorkeur aan een één op één gesprek met een domeinexpert. Ook twee op één komt vaak voor. Ze vindt het prettig als ze niet teveel mensen tegenover zich heeft, omdat deze al gauw door elkaar gaan roepen. Af en toe wordt er een workshop gehouden, maar daarbij moet opgepast worden dat er geen mensen gaan overheersen, of dat bepaalde mensen niet uit de verf komen. Ook komt het voor dat de sfeer in een bedrijf niet zo goed is, en dat er daardoor politieke discussies ontstaan in plaats van inhoudelijke.

Om te zorgen dat ze de juiste mensen te spreken krijgt, overlegt mevr. Bleeker met de opdrachtgever welke mensen ze nodig heeft. Ze geeft daarbij aan dat ze er van op aan wil kunnen dat deze mensen ook doen wat ze zeggen. Meestal zijn dit de mensen die het het allerdrukst hebben en vaak geen tijd hebben, dus in sommige gevallen moet ze genoegen nemen met mensen die wat minder goed zijn. Hier is vaak niks aan te doen, behalve het aangeven aan de opdrachtgever welk risico hij loopt wanneer de beste mensen niet beschikbaar zijn. Het is belangrijk dat deze mensen ook door hun collega's gewaardeerd worden. Mevr. Bleeker ervaart het als prettig als deze experts zorgen voor hun eigen achterban. Op het moment dat ze met deze mensen praat, zorgen zij zelf dat alles wat zij zeggen uitgelegd en afgestemd wordt met de achterban. Dit voorkomt dat mevr. Bleeker met iedereen in discussie moet gaan. Ook is het belangrijk dat ze naast experts ook met mensen praat die beslissingen kunnen nemen, zodat het niet gebeurt dat ze inhoudelijk een goed verhaal heeft, maar niemand een beslissing kan nemen.

Voordat mevr. Bleeker met modelleersessies begint, stelt ze eerst het doel vast van het model. Daar hoort ook bij op welke manier ze het gaat documenteren. Ze heeft daar een aantal standaardmanieren voor waarvan ze weet dat die voor haar goed werken, maar als er bijvoorbeeld haast is in een project, is creativiteit vereist. Hierbij is ook van belang of mensen de modellen kunnen lezen of niet, en hoe de documentatie verspreid zal worden.

Het aantal aanwezigen in een modelleersessie hangt af van de doelstelling van die sessie. Wanneer het bijvoorbeeld gaat om de afbakening van de scope, dan is het van belang dat alle vertegenwoordigers er zijn. Wanneer er echter gesproken wordt over een specifiek onderwerp, dan zijn er een stuk minder mensen nodig, een stuk of drie experts is dan vaak genoeg. Ook de tactiek kan per modelleersessie verschillen. Bij een grote groep kan het handig zijn om mensen dingen op te laten schrijven, maar soms spreekt mevr. Bleeker de mensen één voor één aan. Ze geeft daarbij zelf aan wat ze eerst wil horen, bijvoorbeeld om eerst overzicht te krijgen. Het is belangrijk dat mensen niet door elkaar gaan schreeuwen. Wanneer er punten genoemd worden die op dat moment nog niet van belang zijn, kunnen deze bijvoorbeeld opgeschreven worden, zodat deze punten geparkeerd worden.

Door zowel de interviews als de workshops te starten met een zin als “Ik weet niks, leg het me maar uit”, zorgt mevr. Bleeker ervoor dat ze niet gehinderd wordt door enige voorkennis, en dat ook informatie naar boven komt die voor de experts triviaal is of waarvan ze zich niet realiseren dat die informatie belangrijk is voor een IT-systeem. Als na wat doorvragen enigszins duidelijk is hoe het domein in elkaar zit, kan er ook gekeken worden of bepaalde processen wel optimaal worden uitgevoerd, en of ze niet overbodig zijn. Daarnaast let mevr. Bleeker erop of het logisch en consistent is wat haar verteld wordt.

Wanneer iemand iets gezegd heeft, creëert mevr. Bleeker voor zichzelf een beeld, en vervolgens gaat ze aan de andere aanwezigen vragen of dat beeld klopt, en of zij het er mee eens zijn. Het kan voorkomen dat mensen het ergens niet mee eens zijn. Door middel van doorvragen probeert mevr. Bleeker dan de oorzaak hiervan te achterhalen, bijvoorbeeld door te vragen wat die persoon als probleem verwacht. Vaak blijkt dan toch iets niet duidelijk te zijn, en zodra het helder is, dat deze persoon dan toch met de oplossing kan leven. Als mensen iets echt niet willen, dan doet mevr. Bleeker vaak even twee stappen terug doen, om met iets simpelers een basis te leggen, zodat ze aan het idee kunnen wennen. Dit heeft volgens mevr. Bleeker ook te maken met het ambitieniveau. Mensen bedenken vaak allerlei oplossingen, die in de praktijk niet altijd haalbaar zijn, bijvoorbeeld door gebrek aan tijd en geld. Het kan dan helpen om een ambitieniveau vast te stellen. Als mensen meteen aan een fantastische oplossing denken, dan probeert mevr. Bleeker een weg te vinden naar iets waar de mensen genoegen mee nemen.
Wanneer ze er echt niet uit komt, dan wordt de betreffende issue voorgelegd aan iemand die een knoop daarover kan doorhakken, omdat de discussies anders eindeloos worden.

Als er onenigheid is wat betreft naamgeving dan zijn er verschillende oplossingen mogelijk. Soms kiest mevr. Bleeker zelf een oplossing waar de mensen zich op een gegeven moment in kunnen vinden. In het ergste geval kan er gekozen worden om meerdere termen te gebruiken, als het echt heel belangrijk is dat het onderscheid tussen die termen er is. Het belangrijkste hierbij is echter dat het begrip hetzelfde is. Als een bepaald begrip bij iemand iets heel anders betekent dan bij iemand anders en diegene heeft een heel ander beeld wat de invulling van dat begrip is, dan levert dat een heel ander soort model op.

Mevr. Bleeker stopt met modelleren als ze zelf het gevoel heeft dat ze het overzicht heeft. Vaak wordt dit echter bepaald door het ambitieniveau. Als er weinig tijd is moet ze in een korte tijd zoveel mogelijk zien te bereiken, en kan ze het niet altijd zo ver uitwerken als ze zou willen. Wat mevr. Bleeker in zo’n geval meestal doet, is, als ze bijvoorbeeld een UML plaat maakt, minimaal alle entiteiten daarin verwerken. Ze vindt het dan niet nodig om ook alle attributen er in te zetten. Dit verschilt echter ook per project.

Mevr. Bleeker heeft een set van documenten die ze tijdens een project altijd maakt. Ze heeft altijd een domeinmodel, al is het maar op een kladje. Vaak heeft ze ook documenten met uses cases en non functional requirements. Tevens worden er tijdens een project regelmatig memo's geschreven om uit te leggen hoe iets in elkaar zit, of wat bijvoorbeeld in een volgende release zit. De meeste documenten zijn ook voor de opdrachtgever, maar er zijn ook altijd een paar dingen die ze voor zichzelf bewaart, zoals haar aantekeningen of lijstjes met issues of dingen die ze nog moet doen. Die documenten gebruikt mevr. Bleeker nog wel eens voor volgende projecten. Zeker dingen als inleidingen, waar bijvoorbeeld in uitgelegd wordt wat een domeinmodel is, gebruikt ze nog wel eens. Omdat ze veel in verschillende domeinen werkt, kan er van de modellen niet veel hergebruikt worden. Er zijn echter wel wat standaard functionaliteiten, zoals logins, autorisaties en beheergedeeltes die wel eens terug komen, en die eventueel hergebruikt kunnen worden.

4.3.2 Koppeling naar de modellen

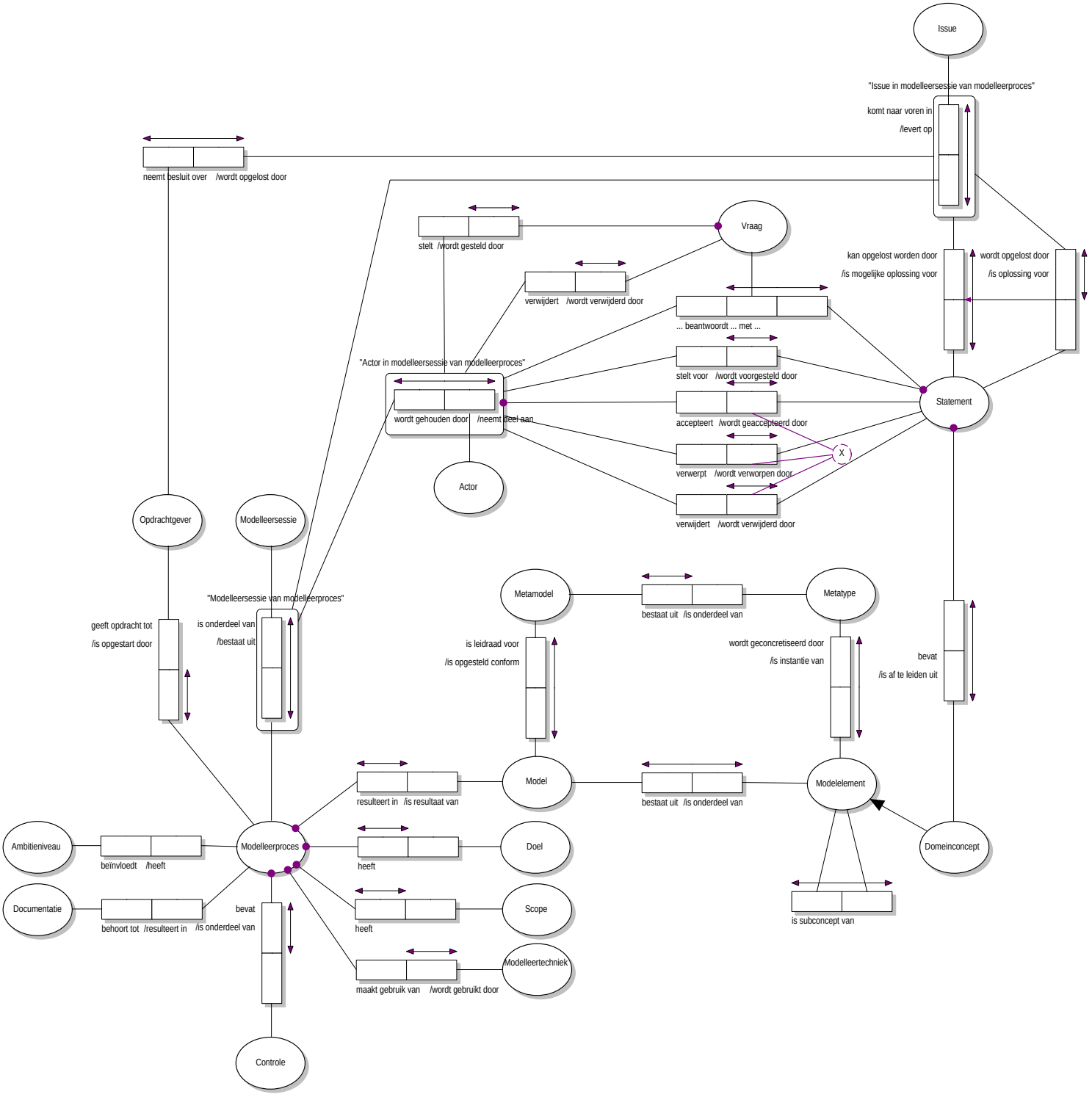
Het modelleerproces dat mevr. Bleeker volgt, is vergelijkbaar met het proces dat reeds in paragraaf 2.4 beschreven is, en dat aangepast is naar aanleiding van het interview met dhr. De Vries. De werkwijze van mevr. Bleeker wordt hieronder vergeleken met de modellen.

Voordat mevr. Bleeker met een modelleersessie begint, stelt ze naast het doel van het model ook vast hoe ze het gaat documenteren. In zowel het ORM model als de flowchart is momenteel nog niets terug te vinden over documentatie. Gezien het feit dat de wijze van documenteren voor iedere persoon anders zal zijn, en niet iedere persoon van te voren bepaalt hoe hij gaat documenteren, zal het vaststellen van de documentatiewijze niet toegevoegd worden als stap in het modelleerproces. Het documenteren zelf zal wel als stap toegevoegd worden, omdat dit in ieder project voor zal komen, zij het op verschillende manieren. De documentatie wordt gezien als behorende bij het modelleerproces, en niet slechts bij het model, omdat dingen als non functional requirements niet direct documentatie zijn van het model, maar wel in het proces opgeleverd kunnen worden.

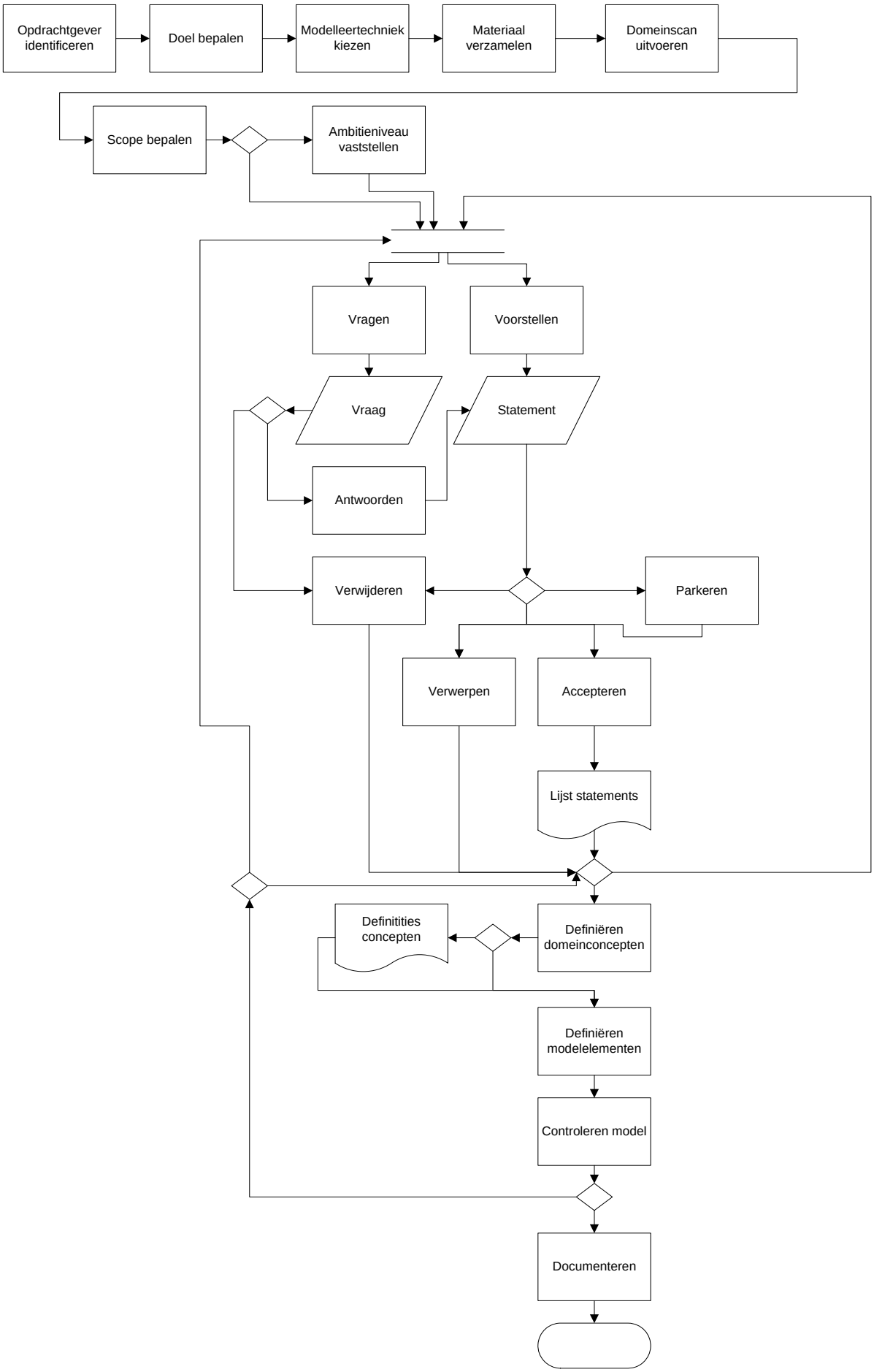
Het parkeren van issues zoals mevr. Bleeker dat beschrijft, komt reeds terug in het model. Ook het feit dat issues voorgelegd worden aan iemand die daar over kan beslissen, wanneer er geen oplossing gevonden kan worden, is in het model verwerkt.

Mevr. Bleeker noemt bij het modelleerproces ook het begrip ambitieniveau. Hiermee duidt ze aan dat niet altijd de ideale oplossing gerealiseerd kan worden, bijvoorbeeld vanwege gebrek aan tijd en geld. Hiermee moet reeds bij het modelleren rekening gehouden worden. Het vaststellen van het ambitieniveau zal toegevoegd worden aan de modellen. In de flowchart wordt dit geen verplichte stap, omdat in de praktijk het ambitieniveau niet altijd expliciet vastgesteld zal worden. Ook in het ORM model zal ambitieniveau geen verplichte rol spelen. Het is echter zodanig van belang bij een modelleerproces, dat het wel meegenomen wordt in de modellen.

De nieuwe versies van de modellen zijn te vinden op de volgende bladzijden.



Figuur 4.7: ORM model versie 0.5



Figuur 4.8: Flowchart versie 0.4

4.4 Interview dhr. Crompvoets

4.4.1 Algemeen verslag

De derde persoon die werd geïnterviewd was dhr. Crompvoets. Hij werkt vanaf 1985 in de ICT en is mede-oprichter van Bommeljé Crompvoets en partners bv. Als adviseur zet en houdt hij automatiseringsprojecten op het juiste spoor. Zijn focus ligt in het toepassen van architectuurprincipes voor het vervullen van lange termijn organisatiedoelstellingen [BCP]. Momenteel is dhr. Crompvoets bezig met een groot project voor de Belastingdienst.

Voor het maken van modellen gebruikt dhr. Crompvoets verschillende technieken, zoals FCO-IM en ER diagrammen. De keuze van de techniek wordt aangepast aan de organisatie waar het project wordt uitgevoerd, zo wordt bij de Belastingdienst gebruik gemaakt van een eigen methode, genaamd BMI. Ook het doel en het niveau van het model hebben invloed op de keuze voor een bepaalde techniek.

De aanpak die dhr. Crompvoets volgt hangt af van de situatie. Hij ziet de werkelijkheid als een soort vage wolk, en probeert daar met een groep mensen duidelijkheid in te krijgen en begrippen op elkaar af te stemmen. Er wordt dan vaak gebruik gemaakt van modelleersessies waarbij geprobeerd wordt om tot een soort gedeeld beeld te komen. In een groot project gebruikt dhr. Crompvoets deze sessies als afstempunt, en gaat tussendoor delen uitwerken om het helder te krijgen. In een kleiner project is zo'n sessie meer een creatief proces om tot het model te komen. Zes à zeven personen is bij een sessie over het algemeen het maximum, waarbij een goede rolverdeling nodig is; er moet een moderator zijn, en iemand die het allemaal opschrijft. In theorie zou er ook altijd iemand aanwezig moeten zijn die beslissingen kan nemen, maar in de praktijk haakt deze vaak af. Dhr. Crompvoets wil liever niet meer dan een stuk of drie materiedeskundigen, om te voorkomen dat deze elkaar gaan tegenspreken. Wanneer het echter nodig is om in een groter bedrijf draagvlak te creëren dan ligt het aantal aanwezigen een stuk hoger.

Wanneer hij al langer in een organisatie rondloopt, wijst dhr. Crompvoets zelf aan welke mensen hij in een modelleersessie wilt hebben. Dit is echter niet mogelijk wanneer hij de organisatie nog niet kent. Hij krijgt dan vaak een lijstje namen van de opdrachtgever, maar de beste mensen hebben het altijd druk en zijn hierdoor niet altijd beschikbaar. Dhr. Crompvoets heeft ervaren dat hij er altijd zelf erg aan moet trekken om een en ander van de grond te krijgen, omdat modelleersessies moeilijk te plannen zijn.

Voordat een modelleersessie begint, bestudeert dhr. Crompvoets vaak al documenten. Dit is vooral het geval wanneer hij al bekend is in de organisatie, omdat hij dan meer toegang hiertoe heeft. Om het doel van een sessie te kunnen bereiken, probeert hij zich zoveel mogelijk voor te bereiden en zoveel mogelijk in te lezen om het proces goed te kunnen sturen. Als het al duidelijk is welke richting hij op wil, vindt dhr. Crompvoets het makkelijker om te focussen in een sessie. De kans is groot dat de aanwezigen anders over iets gaan praten wat niet het probleem is. Wanneer er direct over het juiste onderwerp gepraat wordt, hoeft dhr. Crompvoets mensen minder bij te sturen en af te kappen, wat een positieve invloed heeft op het draagvlak. Bij de kick off van een project is het voorbereiden moeilijker, en gaat dhr. Crompvoets meer blanco een sessie in.

Dhr. Crompvoets begint een sessie altijd met een voorstelrondje. Daarna probeert hij een beeld te krijgen van de strategie en het toekomstbeeld van het bedrijf, omdat als er nieuwe oplossingen worden bedacht, deze voor het bedrijf zijn zoals dat er over twee jaar uit ziet. Vervolgens zoomt hij in op de producten en diensten die bij het bedrijf horen. Hier probeert hij steeds verder op in te zoomen, hij pelt het product of de dienst als het ware af, vanuit de verschillende views van de aanwezigen. Als voor iedereen helemaal duidelijk is wat het begrip inhoudt, dan kan hetzelfde gedaan worden met een volgend begrip. Het kan ook voorkomen dat daarbij nieuwe begrippen geïntroduceerd worden, die de aanwezigen in hun dagelijkse werkzaamheden nog niet gebruiken. Op die manier kunnen begrippen gekoppeld worden, door een abstractieniveau hoger te gaan.

Discussies in modelleersessies ontstaan volgens dhr. Crompvoets vaak wanneer mensen langs elkaar heen praten of een term net iets anders interpreteren. Het kan zijn dat ze het in feite helemaal met elkaar eens zijn, maar omdat ze iets niet helemaal begrijpen, het in een sessie volledig met elkaar oneens zijn. Hij probeert dit zelf altijd te sturen, en begrippen volledig uit te pellen zodat ze helder worden. Hij doet dit door veel vragen te stellen om duidelijk te krijgen wat mensen onder een begrip verstaan. Op die manier kunnen issues opgelost worden die zijn ontstaan door begripsverwarring. Wanneer iemand zich in een hoek gedrukt voelt in de groep, of wanneer één iemand heel dominant is, dan ontstaan er politieke issues. Diegene heeft zich bijvoorbeeld vastgebeten in een standpunt en wil niet toegeven omdat hij bang is voor gezichtsverlies. Ook hier is het belangrijk om veel te praten, en begrippen duidelijk te maken. Er moet vooral niet op een te abstract niveau gepraat worden, want dan heeft iedereen in feite gelijk.

Wanneer uit een sessie iets niet geheel duidelijk is geworden, dan gaat dhr. Crompvoets achteraf mensen één op één benaderen. Dit gebeurt echter ook wel eens vóór een sessie. Als dhr. Crompvoets van tevoren al een idee heeft wat hij uit een bepaalde sessie wil halen, dan sorteert hij de mensen daar op voor, en probeert hen soms al bepaalde ideeën in het hoofd te planten, als onderdeel van het politieke spel.

In een volgende sessie wordt er gefocust op dingen die nog onduidelijk zijn. Dhr. Crompvoets behandelt dan geen begrippen meer die al duidelijk waren, omdat de kans dan groot is dat er nieuwe discussies ontstaan, waardoor het risico aanwezig is dat een sessie opnieuw gedaan wordt, terwijl er juist dieper gegaan moet worden. Het is belangrijk om op de blinde vlekken te focussen, zeker omdat er in een sessie altijd minder gedaan wordt dan dhr. Crompvoets eigenlijk wil, omdat iedereen de ruimte nodig heeft om te praten. Door van te voren dingen al één op één met mensen af te stemmen, is het niet nodig om een open vraag te stellen, en stuurt dhr. Crompvoets mensen al een bepaalde richting op.

Wanneer een model af is, hangt volgens dhr. Crompvoets af van de doelstelling. Wanneer er een echte projectdoelstelling is, is het af wanneer er voldoende concrete specificaties zijn om te kunnen gaan bouwen. Bij een sessie die meer ad hoc gehouden wordt, stopt dhr. Crompvoets als er een gedeeld beeld is en iedereen het gevoel heeft dat het een goede sessie was, het gaat dan meer om een groepsproces.
Dhr. Crompvoets benadrukt dat een model bijna nooit af is en dat veel bedrijven er daarom weinig rendement uit halen. Ze zien het als eenmalige actie, terwijl het juist iets is dat mee moet evolueren met het bedrijf. De aanpassingen in het model zouden moeten leiden tot aanpassingen in de systemen.

Doordat in de sessies continu vragen worden gesteld om de verschillende gezichtspunten te valideren, hoeft dit niet achteraf nog gedaan te worden. Dhr. Crompvoets vindt dat elk model een bepaalde elegantie of esthetiek in zich heeft. Hij ziet vaak aan de vorm al of het goed is of niet, zonder te weten wat het onderwerp is. Wanneer alle lijnen kris kras door elkaar heen lopen, en alles met alles verbonden is, is dat voor hem een teken dat de maker ervan het gebied niet goed doorzien heeft.

In de documentatie komt vaak niet het hele model te staan, omdat dit veel te uitgebreid en niet hanteerbaar is. Er worden wel vaak delen van modellen gebruikt, omdat volgens dhr. Crompvoets iedereen denkt in blokjes, rondjes en pijltjes, en het dus een goed communicatiemiddel is. Het belangrijkste is dat de juiste begrippen er in terug komen, er moet een bepaalde kernboodschap in zitten.

4.4.2 Koppeling naar de modellen

Het modelleerproces dat dhr. Crompvoets beschrijft, komt overeen met het modelleerproces zoals dat af te leiden is uit de modellen.

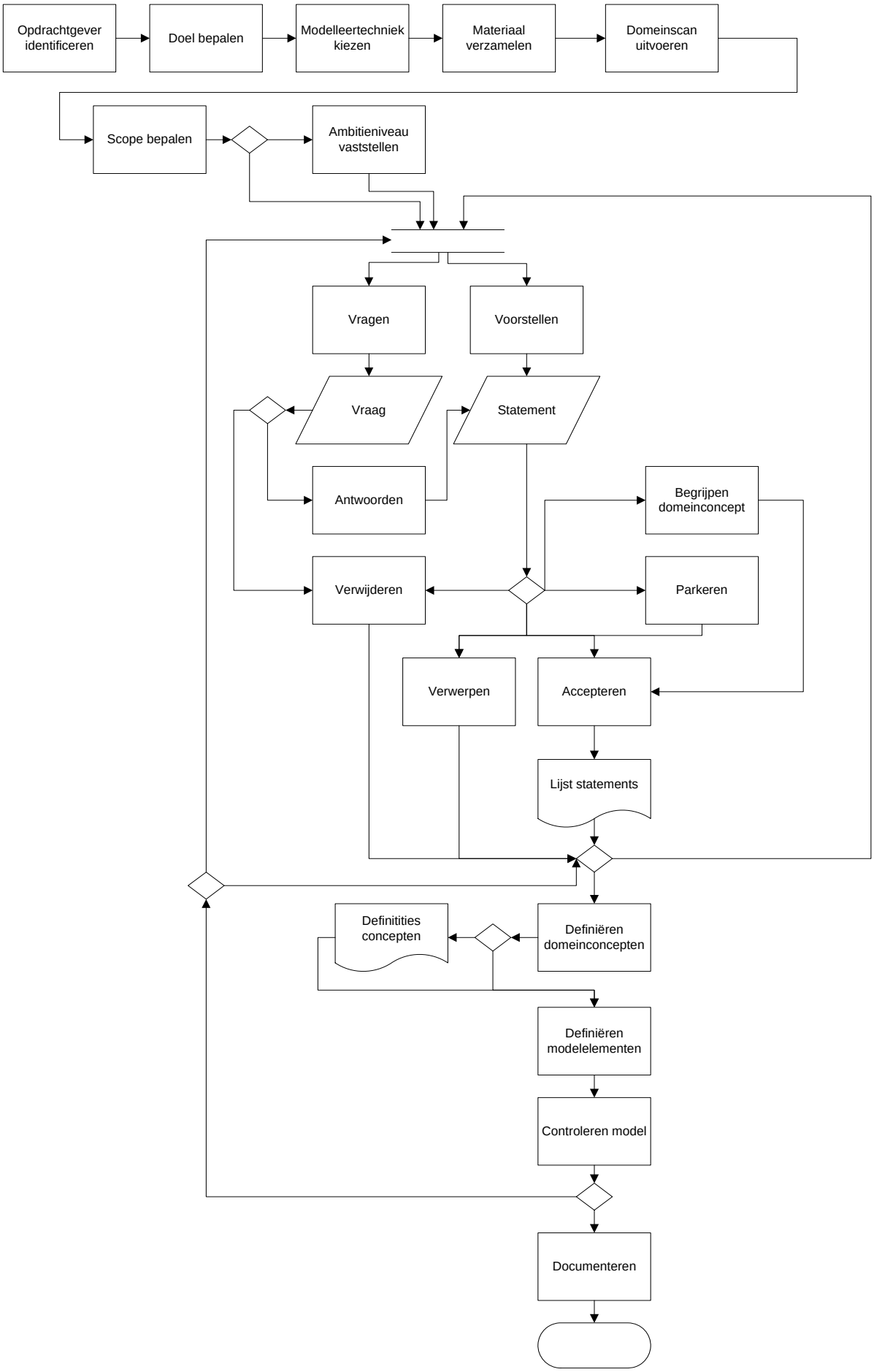
Voordat dhr. Crompvoets met een modelleersessie begint, stelt hij eerst het doel vast, zodat hij vervolgens een modelleertechniek kan kiezen. Wanneer hij de kans heeft, verzamelt hij al voor een sessie zoveel mogelijk documenten om zich voor te bereiden en in te lezen. Deze stap komt overeen met het uitvoeren van een domeinscan.

Dhr. Crompvoets start een modelleersessie vaak met het helder krijgen van het toekomstbeeld en de strategie van de organisatie. In de modellen is dit niet terug te vinden als eerste stap in een modelleersessie. Niet alle modelleurs beginnen daar hun sessie mee. Het toekomstbeeld kan reeds duidelijk geworden zijn bij het identificeren van de opdrachtgever, of bij het uitvoeren van een domeinscan. Dit is een stap die ieder op zijn eigen manier uitvoert, en op de plaats in het proces waar het hem het beste uitkomt. Gezien het feit dat deze stap gecombineerd kan worden met stappen die reeds in de modellen terug te vinden zijn, wordt hier geen nieuwe stap voor gecreëerd.

Wat ook in het interview met mevr. Bleeker naar voren kwam, is dat er een bepaalde rolverdeling moet zijn binnen de modelleersessies. Dhr. Crompvoets praat over een moderator en iemand die alles noteert. Ook iemand die beslissingen kan nemen kan aanwezig zijn bij zo'n sessie. Deze verschillende rollen zijn niet terug te vinden in de modellen. In het ORM model zou dit opgelost kunnen worden door het toevoegen van subtypes aan "Actor". Echter, het is niet nodig om extra informatie op te slaan over deze soorten actoren, waardoor het niet nodig is om deze ook daadwerkelijk aan het model toe te voegen.

Uit het interview blijkt eveneens hoeveel het stellen van vragen en het definiëren van concepten met elkaar te maken heeft. Om tot een goede definitie van een concept te komen, is het van belang dat er geen aannames worden gedaan, maar dat het begrip compleet helder is. Dit kan alleen bereikt worden door het stellen van enorm veel vragen. Het hebben van een duidelijk beeld van de betekenis van een concept, kan bijdragen tot de acceptatie van een statement. Uit de flowchart is dit niet duidelijk af te leiden. Er wordt daarom een proces "Begrijpen domeinconcept" toegevoegd, die wordt verbonden met "Statement". Een vraag wordt in de modelleersessie immers beantwoord met een statement. Het begrijpen van een domeinconcept leidt vervolgens tot acceptatie van een statement. Daarom wordt "Begrijpen domeinconcept" verbonden met "Accepteren". "Definiëren domeinconcepten" blijft als proces ongewijzigd, om het onderscheid te behouden tussen het helder hebben wat een concept inhoudt, en het opstellen van een definitie van het concept. Het opstellen van een definitie gebeurt pas wanneer voor alle aanwezigen in de sessie helder is wat het concept inhoudt.

De nieuwe versie van de flowchart is te vinden op de volgende pagina. Het ORM model blijft ongewijzigd.



Figuur 4.9: Flowchart versie 0.5

5. Slot

De probleemstelling zoals deze in de inleiding gepresenteerd werd, was als volgt:

“Hoe zien het ORM model en het procesmodel eruit waarmee modelleerprocessen en hun verloop vastgelegd kunnen worden?”

Deze probleemstelling was onderverdeeld in een aantal onderzoeksvragen, die eveneens in de inleiding te lezen zijn. Deze vragen zijn bij de totstandkoming van de modellen reeds beantwoord. Een uitgebreide beantwoording van de onderzoeksvragen zou leiden tot een herhaling van de informatie die in deze scriptie te vinden is. Daarom wordt er slechts een kort antwoord gegeven op de onderzoeksvragen, waarbij verwezen zal worden naar de plaats in de scriptie waar meer informatie te vinden is.

5.1 Beantwoording van de onderzoeksvragen

De onderzoeksvragen O1 en O2 hebben geresulteerd in twee modellen, te weten een ORM model en een procesmodel. Deze twee modellen samen zijn tevens het antwoord op de probleemstelling. De modellen zullen bij paragraaf 5.2 “Beantwoording van de probleemstelling” weergegeven worden, wat betekent dat deze twee onderzoeksvragen in deze paragraaf buiten beschouwing worden gelaten.

5.1.1 Beantwoording onderzoeksvraag O3

Onderzoeksvraag O3 bestaat uit drie deelvragen, te weten:

O3a: Uit welke activiteiten bestaat een modelleerproces?

O3b: Welke rol spelen deze activiteiten binnen het modelleerproces?

O3c: In welke volgorde worden deze activiteiten uitgevoerd?

Gezien het feit dat deze vragen een grote samenhang vertonen, worden deze tegelijkertijd beantwoord.

Binnen een modelleerproces wordt allereerst een opdrachtgever geselecteerd. Deze opdrachtgever is van belang in het modelleerproces wanneer er mensen geselecteerd moeten worden voor een modelleersessie, maar bijvoorbeeld ook wanneer er knopen doorgehakt moeten worden. Wanneer het doel van het model bepaald is, kan er gekozen worden voor een modelleertechniek, die past bij het doel. In de meeste gevallen wordt er vervolgens materiaal verzameld waarmee een domeinscan uitgevoerd kan worden. Dit materiaal kan bestaan uit bijvoorbeeld probleembeschrijvingen, uitvoerformulieren van reeds bestaande applicaties, of een gesprek met de opdrachtgever. In de domeinscan wordt het materiaal geanalyseerd, en worden onder andere de stakeholders geïdentificeerd. Ook kan de domeinscan dingen duidelijk maken wat betreft de scope van het modelleerproces. Het vaststellen van de scope is dan ook de volgende stap. De volgende stap is het vaststellen van het ambitieniveau, die overigens niet in elk modelleerproces expliciet wordt genomen. In deze stap wordt gekeken op welk niveau de oplossing gecreëerd zal worden, want aangezien er nooit genoeg tijd en geld is, kan er vrijwel nooit een perfecte oplossing gerealiseerd worden.
Dit deel van het modelleerproces is uitgelegd in paragraaf 2.4. In de paragrafen 4.1, 4.3 en 4.4 wordt de theorie aan de praktijk gekoppeld door middel van interviews.

Het tweede deel van het modelleerproces betreft de modelleersessies die gehouden worden met stakeholders. Hier vallen zowel één op één gesprekken onder als sessies in een groep. In deze fase wordt het domein doorgespit door het stellen van vragen en het doen van voorstellen. Door het beantwoorden van vragen en het accepteren en verwijderen van voorstellen, wordt het gebied helder voor de systeemanalist. De theorie over dit onderdeel wordt eveneens in paragraaf 2.4 behandeld. Hoe modelleersessies in de praktijk verlopen, kan afgeleid worden uit de interviews die in de paragrafen 4.1, 4.3 en 4.4 beschreven staan. In paragraaf 4.2 worden twee voorbeelden van modelleersessies getoond.

Wanneer uit de sessies alle concepten duidelijk zijn geworden, kunnen deze domeinconcepten gedefinieerd worden. Vervolgens wordt het model opgesteld, door het definiëren van modelelementen. Meer hierover is te vinden in paragraaf 4.2.2. Als laatste zullen er nog enkele controles uitgevoerd worden, en wordt de documentatie in orde gemaakt. Ook hiervoor geldt dat de theorie te vinden is in paragraaf 2.4 en de praktijk af te leiden is uit de interviews.

5.1.2 Beantwoording onderzoeksvraag O4

Onderzoeksvraag O4 is gesplitst in twee deelvragen, namelijk:

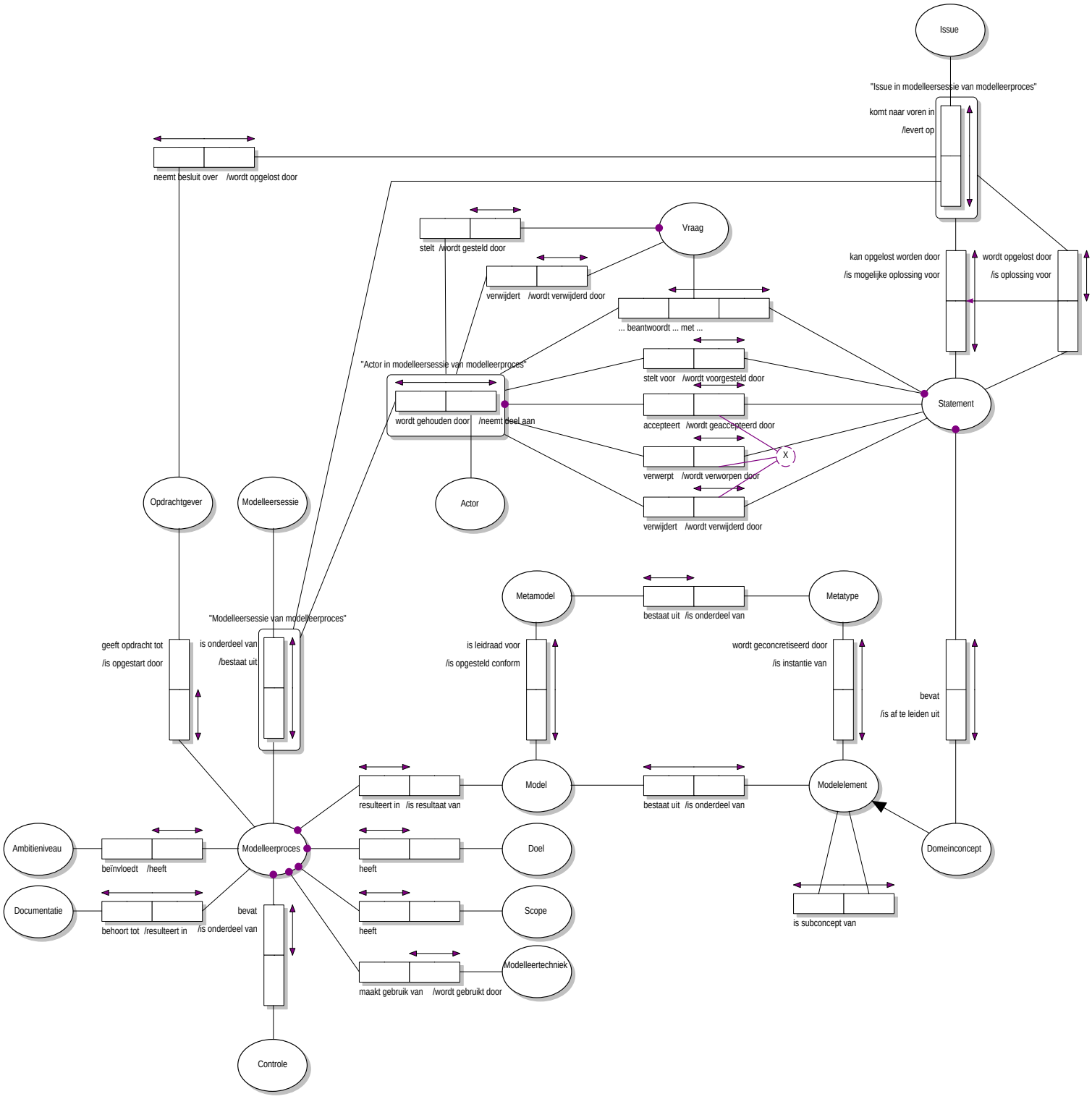
O4a: Uit welke elementen bestaat een model?

O4b: Hoe komen deze elementen voort uit het modelleerproces?

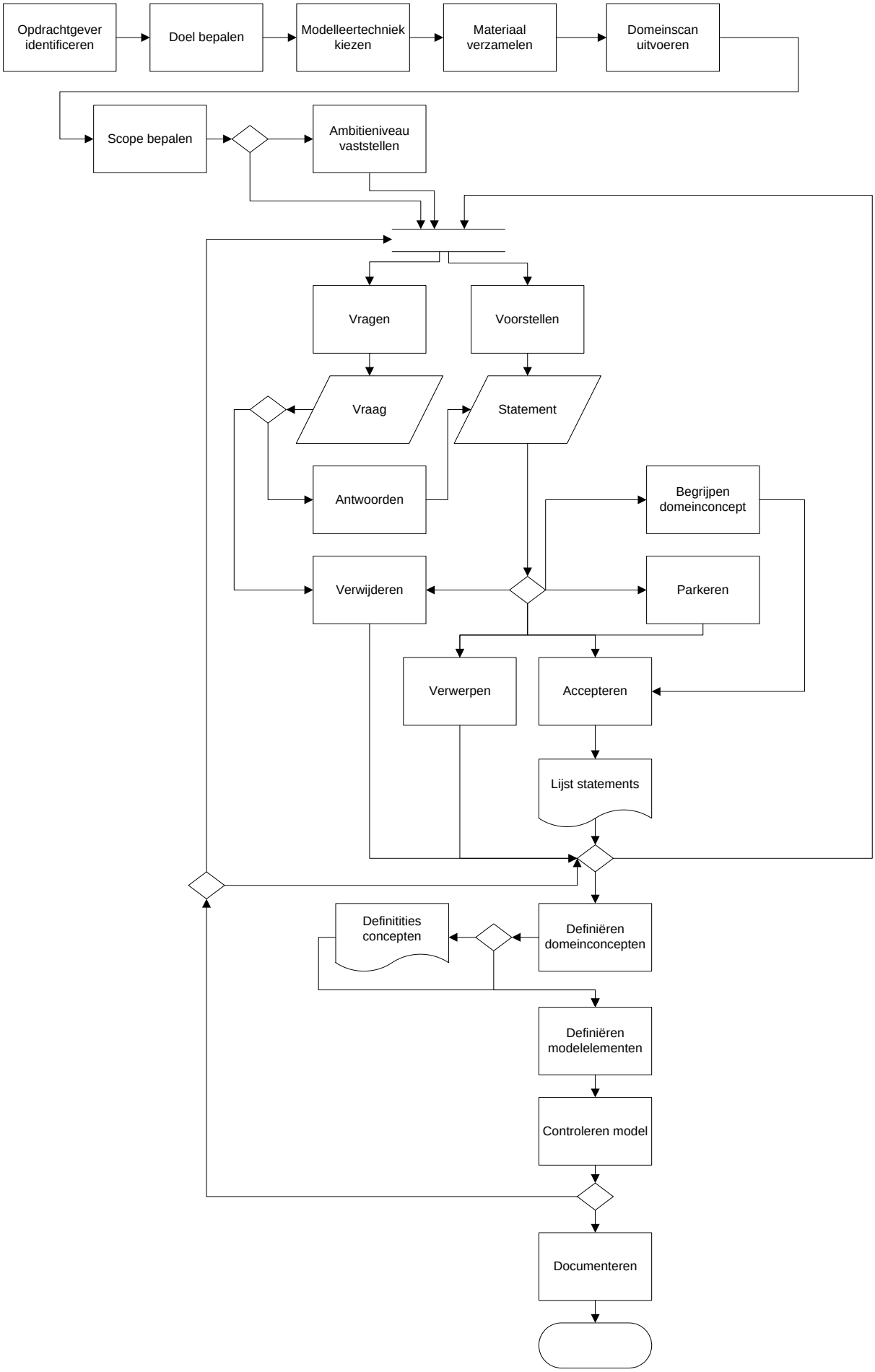
Ook deze vragen zullen in één keer beantwoord worden. In de paragrafen 2.1, 2.2 en 2.3 worden werkwijzen beschreven voor respectievelijk ORM, UML klassendiagrammen en ER diagrammen. Hieruit wordt al duidelijk dat elke techniek zijn eigen modelelementen kent. In paragraaf 2.4 worden de werkwijzen gegeneraliseerd, en wordt er onderscheid gemaakt tussen concepten, relaties en constraints, omdat deze in alle drie de werkwijzen naar voren komen. In hoofdstuk 3, de validatie van het ORM model, blijkt echter dat dit niet specifiek genoeg is, en wordt er rekening gehouden met attributen, identificatoren, datatypen, relatienamen, subtypen, cardinaliteiten en verplichte rollen. Dit is goed te zien in ORM model versie 0.3.
Zelfs deze uitbreiding is niet voldoende specifiek. In paragraaf 4.2.2 is besloten om dit niet verder uit te diepen, om te voorkomen dat twee studies naast elkaar zouden gaan lopen. Er is een constructie geïntroduceerd die gebruik maakt van metamodellen, metatypen en modelelementen.

5.2 Beantwoording van de probleemstelling

Het antwoord op de probleemstelling “Hoe zien het ORM model en het procesmodel eruit waarmee modelleerprocessen en hun verloop vastgelegd kunnen worden?” bestaat uit twee modellen. De definitieve versies van de modellen zullen op de volgende bladzijden weergegeven worden.



Figuur 5.1: Definitieve versie ORM model



Figuur 5.2: Definitieve versie flowchart

5.3 Beperkingen en aanbevelingen

Zoals reeds in paragraaf 4.2.2. en bij de beantwoording van onderzoeksvraag O4 is vermeld, is het gedeelte binnen het ORM model waar de modellen opgeslagen worden, niet ideaal. In de toekomst zou het ORM model omgezet kunnen worden in een tool, waarmee de modelleerprocessen en de resulterende modellen opgeslagen kunnen worden. Het verdient dan aanbeveling om nog aandacht aan dit gedeelte van het model te besteden, zodat de modellen zonder problemen opgeslagen kunnen worden.

Het gedeelte in het ORM model waar de modelleerprocessen mee opgeslagen kunnen worden, is gevalideerd aan de hand van interviews. Uit de interviews zijn uitbreidingen op de modellen naar voren gekomen. Er zijn echter geen tegenstrijdigheden gebleken binnen deze interviews, wat mijns inziens een teken is dat de opgestelde modellen een goede basis bieden voor een mogelijke tool. Het zal waarschijnlijk minder aanpassing vereisen wanneer deze tool gebouwd zal worden dan het bovengenoemde gedeelte. Echter, in de praktijk zullen er waarschijnlijk nog kleine aanpassingen naar voren komen, wanneer daadwerkelijk geprobeerd wordt om modelleerprocessen op te slaan. Zeker omdat ieder modelleerproces anders is, is het moeilijk om een standaard modelleerproces te definiëren. De aanpassingen die er mogelijkerwijs komen, zullen vooral kleine uitbreidingen betreffen, bijvoorbeeld zaken die bepaalde personen op willen slaan tijdens een modelleerproces, maar die niet standaard door elke modelleur opgeslagen worden.

De aanpak die ik heb gevolgd om tot de modellen te komen, is het bewust niet in de breedte kijken naar modelleerprocessen, maar juist systematisch en gericht te werk gaan. Deze aanpak kent zijn voor- en nadelen. Als nadeel kan genoemd worden dat er een weinig theoretisch kader ontstaat. Ook zijn de gebruikte referenties minder divers dan wanneer meer in de breedte naar modelleerprocessen gekeken zou worden.
Het voordeel van deze aanpak is echter dat de weg tot de modellen erg nauwkeurig beschreven kon worden, en dat echt elke stap die genomen is, terug te vinden is. Hierdoor kennen de modellen een goede onderbouwing.

Lijst van figuren

Literatuur

[Amb] Ambler, S. (2002). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. New York: John Wiley & Sons, Inc.

[BCP] Bommeljé Crompvoets en partners bv
http://www.bcp-software.nl

[BW] Bosman, S. & Weide, Th.P. van der (2004). *Towards formalization of the information modeling dialog.* Nijmegen: Computing Science Institute, University of Nijmegen.

[Hal] Halpin, T.A. (2001). *Information Modeling and Relational Databases, From Conceptual Analysis to Logical Design*. San Francisco: Morgan Kaufmann Publishers.

[HBP] Hoppenbrouwers, S., Bleeker, A. & Proper, H. (2004). *Modelling linguistically complex business domains.* Technical report NIII-R0405. Nijmegen, the Netherlands: Nijmegen Institute for Information and Computing Sciences, University of Nijmegen.

[HPM] Hoffer, J.A., Prescott, M. & McFadden, F.R. (2002). *Modern Database Management.* Upper Saddle River, New Jearsy: Prentice Hall.

[HPR05] Hoppenbrouwers, S.J.B.A., Proper, H.A., & Reijswoud, V.E. van (2005). Navigating the Methodology Jungle – The communicative role of modelling techniques in informationsystem development. *Computing Letters,* 1(3).

[HPW05a] Hoppenbrouwers, S.J.B.A., Proper, H.A. & Weide, Th.P. van der (2005). A Fundamental View on the Process of Conceptual Modeling. *Lecture Notes in Computer Science,* volume 3716, 128–143.

[HPW05b] Hoppenbrouwers, S.J.B.A., Proper, H.A. & Weide, Th.P. van der (2005). Formal Modelling as a Grounded Conversation. *Proceedings of the 10th International Working Conference on the Language Action Perspective on Communication Modelling (LAP'05)*, 139–155.

[HPW05c] Hoppenbrouwers, S.J.B.A., Proper, H.A. & Weide, Th.P. van der (2005). Towards explicit strategies for modeling. *Proceedings of the Workshop on Evaluating Modeling Methods for Systems Analysis and Design (EMMSAD'05), held in conjunction with the 17th Conference on Advanced Information Systems 2005 (CAiSE 2005)*, 485–492.

[HPW05d] Hoppenbrouwers, S.J.B.A., Proper, H.A. & Weide, Th.P. van der (2005). Understanding the Requirements on Modelling Techniques. *Lecture Notes in Computer Science*, volume 3520, 262–276.

[HSZ] Hoof, B. van, Seters, H. van & Zwart, J.P. (2002). *Syllabus Proces- en Datamodellering.* Arnhem: Informatica Communicatie Academie, Hogeschool van Arnhem en Nijmegen.

[KO] *Klasse Objecten – Introduction to OCL.*
http://www.klasse.nl/ocl/

[KPH] Kamphuis, V., Proper, H.A. & Hoppenbrouwers, S.J.B.A. (2004). *Informatiekunde visie 2003.* Nijmegen, Nijmeegs Instituut voor Informatica en Informatiekunde.

[Kroen] Kroenke, D.M. (2002). *Databases, Beginselen, ontwerp en implementatie*. Amsterdam: Pearson Education Benelux.

[OMG] *Introduction to OMG's Unified Modeling Language™ (UML®)*.
http://www.omg.org/gettingstarted/what_is_uml.htm

[PBH] Proper, H.A., Bleeker, A.I. & Hoppenbrouwers, S.J.B.A. (2004). Object-Role Modeling as a Domain Modeling Approach. *Proceedings of the Workshop on Evaluating Modeling Methods for Systems Analysis and Design (EMMSAD`04), held in conjunction with the 16th Conference on Advanced Information Systems 2004 (CAiSE 2004).*

[PVH05] Proper, H.A., Verrijn–Stuart, A.A. & Hoppenbrouwers. S.J.B.A.(2005). Towards Utility–based Selection of Architecture–Modelling Concepts. *Conferences in Research and Practice in Information Technology Series,* volume 42, 25– 36.

[PW05] Proper, H.A. & van der Weide, Th.P. (2005). *Modelling as Selection of Interpretation*. Technical report: ICIS-R06004. Nijmegen: Radboud University Nijmegen.

[VHP03] Veldhuijzen van Zanten, G.E., Hoppenbrouwers, S.J.B.A. & Proper, H.A. (2003). System Development as a Rational Communicative Process. *Cybernetics and Informatics*, volume XVI, 126–130.

[Wiki] *Unified Modeling language.* http://nl.wikipedia.org/wiki/Unified_Modeling_Language

[WK] Warmer, J. & Kleppe, A. (2001). *Praktisch UML.* Amsterdam: Adisson Wesley.

Appendix A: Lijst van begrippen

A.1 ORM

Begrippen volgens [Hal].

Afgeleid feittype: Een feittype dat is afgeleid van een ander feittype, gebruikmakend van een afleidingsregel.

Afleidingsregel: Regel die vastlegt hoe een feittype afgeleid kan worden van andere feittypen.

Ariteit: Het aantal rollen in een relatie (unair = 1, binair = 2 etc.).

Conceptueel schema: Conceptueel model van de UoD structuur; ontwerp dat specificeert welke toestanden en veranderingen mogelijk zijn; declaratie van elementaire feittypen, constraints en afleidingsregels.

Constraint: Restrictie op mogelijke toestanden (statische constraint) of veranderingen (dynamische constraint).

Elementair feit: Bewering dat een object een eigenschap heeft, of dat een of meer objecten participeren in een relatie, waarbij het feit niet opgesplitst kan worden in simpelere feiten met dezelfde objecttypen zonder dat er informatie verloren gaat.

Entiteit: Object waarnaar gerefereerd wordt door het te relateren aan andere objecten; geen waarde. Een entiteit mag veranderingen onder gaan over de tijd.

Feit: Een zin die niet gebruikt wordt voor primaire verwijzing. Deze kan elementair of samengesteld zijn.

Feittype: Een soort feit, dat objecttermen en een predikaat bevat.

Generalisatie: Een meer algemeen geval vormen uit meer specifieke typen; het omgekeerde van specialisatie.

Instantie: Een individueel voorkomen; een specifiek lid van een type.

Object: Een ding dat belangstelling wekt. Een object kan een entiteit of een waarde zijn.

Populatie: Een set van instanties die in een bepaalde toestand van de database aanwezig is.

Predikaat: Propositie met objectgaten erin, bijvoorbeeld: "… werkt voor …".

Primitief entiteittype: Entiteittype dat niet afgeleid kan worden uit andere entiteittypen.

Relatie: Eigenschap of associatie die betrekking heeft op een of meerdere objecten.

Rol: Een rol gespeeld door een object in een relatie.

Samengesteld feittype: Een feittype dat equivalent is aan een conjunctie van smallere feittypen.

Specialisatie: Het vormen van speciale gevallen vanuit een meer algemeen type; het omgekeerde van generalisatie.

Subtype: Een objecttype dat deel uitmaakt van een ander objecttype. Subtypen moeten gedefinieerd worden in termen van relaties gespeeld door hun supertype(n).

Type: Set van mogelijke instanties.

UoD: Universe of Discourse; applicatie domein; die aspecten van de wereld waar we over willen praten.

Verwijzing: Relatie die gebruikt wordt als de primaire manier om te refereren naar een object of een object te identificeren.

Verwijzingsmode: Wijze waarop een enkele waarde refereert naar een entiteit; gebruikt om simpele verwijzingschema's af te korten.

Waarde: Onveranderbaar object dat geïdentificeerd wordt door een constante.

A.2 UML klassendiagram

Begrippen volgens [WK].

Associatie: Een associatie is een structurele relatie tussen twee klassen.

Attribuut: Informatie die door een object beheerd wordt. Ieder attribuut heeft precies één waarde voor iedere instantie van de klasse.

Generalisatie: Generaliseren is het classificeren van klassen op grond van identieke kenmerken van deze klassen.

Identifier: Een identifier is de sleutelwaarde van een entiteit, vaak een kunstmatig attribuut, om de entiteit te identificeren.

Instantie: Een instantie is een andere term die voor object gebruikt wordt. Deze term wordt vooral gebruikt om aan te geven in welke klasse een object thuis hoort.

Klasse: Een klasse is een verzameling van objecten met overeenkomstige eigenschappen.

Klassendiagram: Het klassendiagram bevat de klassen in het systeem, hun attributen, operaties en associaties. Het is een model, het beschrijft de structuur waaraan de objecten uit die klassen gebonden zijn.

Link: Een link is een instantie van een associatie.

Modeldictionary: In een modeldictionary worden beschrijvingen van alle klassen opgenomen.

Object: Een object is iets dat een zelfstandig bestaan leidt, in de werkelijkheid of in de geest. Over het algemeen is een object een afspiegeling van ofwel een fysiek ding uit de werkelijkheid, bijvoorbeeld een auto, ofwel een concept uit de werkelijkheid, zoals een bankrekening.

OCL-constraint: OCL staat voor *Object Constraint Language* en is een taal die het mogelijk maakt om expressies en constraints op object georiënteerde modellen te beschrijven.

Operatie: Een operatie beschrijft een service die een object levert. De verzameling operaties van een object representeert het gedrag van het object.

Overerving: Overerving houdt in dat iedere subklasse alle eigenschappen van zijn superklasse erft.

Package: Een package is een groepering van modelelementen, zoals klassen, associaties, generalisaties. Werken met packages is een generiek mechanisme om een opdeling te maken van een softwaresysteem.

Subklasse: Een subklasse is één van de klassen met gezamenlijke kenmerken die opgemerkt zijn bij generalisatie.

Superklasse: De nieuwe klasse die ontstaat wanneer bij generalisatie klassen met gezamenlijke kenmerken samengevoegd worden.

Use case: Eén van de dertien typen diagrammen van UML. Het Use Case diagram toont de actoren en de gebruikersfunctie van het systeem [Wiki].

A.3 ER diagram

Begrippen volgens [HPM].

Attribuut: Een attribuut is een eigenschap of karakteristiek van een entiteittype dat interessant is voor de organisatie.

Cardinaliteit: Het aantal instanties van een entiteit dat geassocieerd kan of moet worden met elke instantie van een andere entiteit.

Entiteit: Een entiteit is een object uit de werkomgeving van de gebruiker waarover de organisatie gegevens wil opslaan.

Entiteittype: Een entiteittype is een verzameling van entiteiten die gemeenschappelijke eigenschappen of karakteristieken bezitten.

Generalisatie: Generalisatie is het proces van het definiëren van een meer algemeen entiteittype van een set van meer gespecialiseerde entiteittypen.

Identifier: Een identifier is een attribuut of combinatie van attributen dat individuele instanties van een entiteittype uniek identificeert.

Relatietype: Een relatietype is een betekenisvolle associatie tussen entiteittypen. “Betekenisvolle associatie” geeft aan dat de relatie het mogelijk maakt om antwoorden te geven op vragen die niet beantwoord kunnen worden als alleen de entiteittypen gegeven zijn.

Subtype: Een subtype is een subgroepering van de entiteiten in een entiteittype die van betekenis is voor een organisatie en die gemeenschappelijke attributen of relaties deelt apart van andere subgroeperingen.

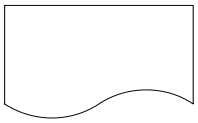
Supertype: Een generiek entiteittype dat een relatie heeft met een of meerdere subtypen.

Specialisatie: Specialisatie is het proces van het definiëren van een of meer subtypen van het supertype en het vormen van supertype/subtype relaties.

Zwak entiteittype: Een zwak entiteittype is een entiteittype die voor zijn bestaan afhankelijk is van een ander entiteittype.

Appendix B: Legenda Flowchart

Proces



Document

Data

Keuze

Parallel mode

Einde