Canonieke data-architectuur

Bert Dingemans

Abstract

Deze whitepaper beschrijft diverse aspecten van canonieke data-architectuur. Naast de definitie van canonieke data-architectuur wordt ingegaan op redenen om een canonieke data-architectuur in te richten.

Canonieke data architectuur kan een bijdrage leveren aan besluitvormingsprocessen. Reden om de politieke aspecten van besluitvorming en verandering te beschrijven. Op basis van vier politieke inrichtingen wordt een handreiking gedaan voor het kiezen van een geschikte werkwijze om een canonieke data-architectuur op te stellen en te valideren.

Het opstellen van canonieke modellen wordt beschreven vanuit verschillende architectuur modelleerwijzen, zoals ArchiMate, ER, berichtenboeken en UML. Er wordt daarnaast op basis van verschillende stakeholders aangegeven welke modelleerwijze toegepast kan worden binnen de organisatie.

Inleiding

Data kent vele verschijningsvormen, het werk van een data-architect bestaat daardoor voor een belangrijk deel uit het in kaart brengen en beschrijven van deze data. Canonieke modellen zijn daarbij van grote waarde. Het doel van canonieke modellen is gericht op het opstellen van een op gestandaardiseerde wijze beschreven register van generieke, herbruikbare (data) objecten.
Canonieke modellen in een data-architectuur kennen echter een groot aantal architectuur dimensies. In dit artikel werken we een aantal van deze dimensies uit vanuit een praktisch perspectief. In de afgelopen jaren hebben canonieke modellen mij geholpen bij het uitwerken van meerdere integratie- en data-architectuur implementaties.

Definitie

Voor canonieke modellen zijn er meerdere definities te vinden. Veelal zie je dat de definities vanuit een bepaald werkveld zijn uitgewerkt. Denk hierbij met name aan SOA en applicatie- of systeemintegratie.
Op Wikipedia is de definitie vanuit integratieperspectief uitgewerkt als een ontwerppatroon gericht op standaardisatie van datadefinities voor de integratie van bedrijfssystemen. Opvallend is hierbij dat men een canoniek model beschouwt als een “dictionary” van herbruikbare gemeenschappelijke objecten. Het canonieke model geldt vervolgens als bron voor discussie tussen technische en bedrijfsmatige betrokkenen.
Deze definitie is voor data-architectuur een goed uitgangspunt, met de opmerking dat het mogelijk te specifiek is uitgewerkt. Met name de beperking naar integratie en naar de interactie tussen techniek en business vind ik te beperkt. Canonieke modellen kunnen gebruikt worden voor het beschrijven van een interface. Vaak zal dit een technische interface zijn (binnen een integratietoepassing) maar ook interfaces van business services, -functies en –processen kunnen uitstekend met een canoniek model beschreven worden. Het beschrijven van de generieke en herbruikbare componenten is vanuit het perspectief van een canoniek model niet noodzakelijk, echter is dat vanuit het gezichtspunt van data-architectuur wel een vereiste. Vandaar dat we dat de volgende definitie formuleren:

Een canoniek model is een beschrijving van herbruikbare, generieke en gestandaardiseerde (data-)objecten binnen een interface. Een interface wordt hierbij beschouwd als een punt van interactie tussen architectuurconcepten.

Waarom data canoniek beschrijven?

Het opstellen van een canoniek model binnen een data-architectuur is een arbeidsintensief traject. Toch besluiten veel organisaties om in deze activiteit te investeren. Daar zijn meerdere redenen voor die allen betrekking hebben op het communiceren van de data definities.
Het communiceren van de datadefinities tussen de stakeholders in de business en betrokkenen uit de ICT met behulp van een canoniek model wordt veelvuldig toegepast. Het canonieke model wordt dan ingezet om de discussie te niet te koppelen van een specifieke technische implementatie maar deze te richten op het generieke functionele en herbruikbare karakter van de data in het model.
Naast deze discussie tussen technici en business is een canonieke data-architectuur ook voor andere vormen van communicatie inzetbaar. Denk hierbij aan de volgende vormen:

- Overeenstemming zoeken bij ketenintegratie tussen twee of meerdere organisaties of afdelingen
- Beschrijven van de data-entiteiten in een datawarehouse of een operational data store
- Documenteren van interfaces van applicaties, services, userinterfaces
- Beschrijven van componenten gerelateerd aan het canonieke model.

Al deze communicatievormen zijn te relateren aan de volgende perspectieven:

- **Ketenintegratie**, het uitwisselen van gegevens tussen twee of meer (autonome) stakeholders neemt? nog steeds in belang toe. Het introduceren van integratie draagt bij aan een efficiëntere informatievoorziening binnen een keten, bedrijfskolom of -cluster.
- **Service-oriëntatie en cloud computing**, binnen deze architectuur worden interfaces en de beschrijving van de data binnen deze interface gebruikt om een maximale ontkoppeling te realiseren en hiermee de betrokken organisaties wendbaarder te maken.
- **Interfacebeschrijving,** veel componenten binnen een architectuur kennen een interface, denk aan gegevensopslag, services, applicaties, componenten en rapportages, maar ook functies, processen en gebruikersinteracties kunnen gebaseerd zijn op datadefinities.

Uit bovenstaande blijkt dat bij het uitwerken van een canoniek model verschillende stakeholders betrokken zijn. Stakeholders met ieder eigen concerns, kennis en agenda. De rol van de data- of informatie-architect is daardoor gericht op het in kaart brengen van betrokkenen, hun belangen en behoeften en hun onderlinge relaties. We zullen dit verder uitwerken onder de term canonieke data-architectuur.

Canonieke data-architectuur

Het beschrijven van de generieke, herbruikbare en gezamenlijke aspecten van gegevens ter ondersteuning van bovenstaande stakeholders is een belangrijke taak binnen de canonieke data-architectuur. Kenmerkend hierbij is dat de communicatie rond de datadefinities gefaciliteerd wordt door de data-architect met onderstaande activiteiten.

Ondersteunen besluitvorming

Voor het realiseren van een canoniek model dat voldoende generiek en herbruikbaar van opzet is, is veel communicatie tussen de betrokken stakeholders noodzakelijk. Vaak blijkt dat er verschillende beelden zijn rond de entiteiten binnen het canonieke model. Denk hierbij bijvoorbeeld aan het homoniemen- en synoniemenprobleem, maar ook aan verschillen in relaties tussen entiteiten of detailleringsverschillen.
In situaties waar de verschillen tussen de beelden groot zijn, zal de discussie met behulp van canonieke data-architectuur ondersteund worden. Hiertoe kan de data-architect kiezen voor verschillende werkvormen. Dit is afhankelijk van hoe de besluitvorming is ingericht (zie hoofdstuk besluitvorming) maar ook de cultuur en structuur van de betrokken organisatie(s) zijn hierbij van belang.
Werkvormen die toegepast kunnen worden zijn interactieve workshops, gezamenlijk modelleren, standaardisatietrajecten of interfacebeschrijvingen. De data-architect heeft hierbij een palet aan vormen tot zijn beschikking en zal afhankelijk van de betrokken stakeholders een keuze maken.

Modelleren

Op basis van bovengenoemde besluitvorming en ter ondersteuning van voorgaande discussies zal de data-architect een canoniek datamodel opstellen. Hierbij zie je veelal dat er verschillende niveaus van modellering ontstaan. Denk hierbij aan een conceptuele, logische en fysieke indeling van de modellen vergelijkbaar met de traditionele datamodellering.
Bij het opstellen van het canonieke model zal rekening worden gehouden met de verschillende stakeholders. Met name kennis, referentiekader maar ook betrokkenheid van de stakeholder stellen eisen aan de verschillende weergaven van de data-architectuur. In het hoofdstuk canonieke data modelleren wordt dit nader uitgewerkt.

Beschrijven implementatie

Op basis van het canonieke model dat opgesteld is binnen voorgaande activiteiten dient de ontsluiting van de entiteiten binnen dit canonieke model geïmplementeerd te worden. Keuzes rond deze implementatie zijn legio. Denk hierbij bijvoorbeeld aan keuzes als inzet van een enterprise service bus, database standaardisatie, maatwerktoepassingen of COTS-toepassingen maar ook de inzet van architectuurrepositories of -registers.

Canonieke data patronen

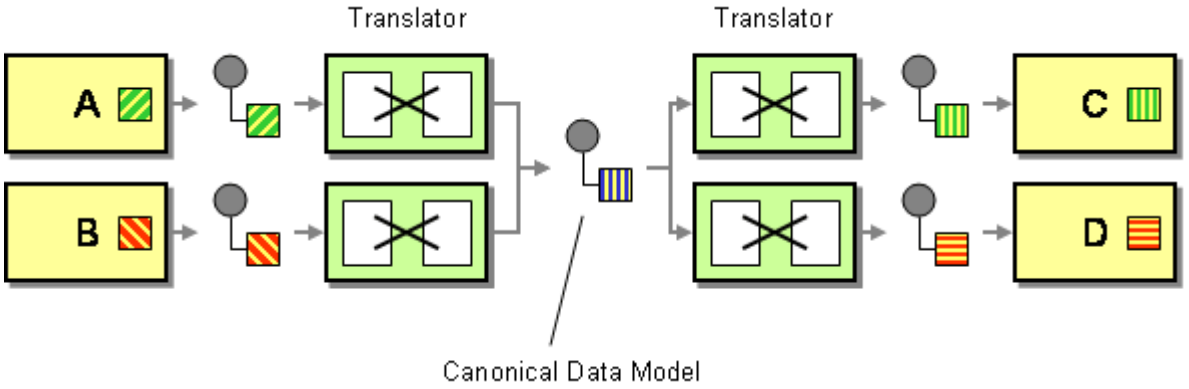
Datapatronen kunnen helpen bij het inrichten van de data-architectuur. Vandaar dat voor het uitwerken van een canonieke data-architectuur reeds aanwezige patronen een kader vormen. In de literatuur zijn een aantal relevante patronen te vinden [Hohpe] [Erl]:

- Canonical data model
- Canonical schema
- Canonical protocol
- Contract centralisation

Canonical data model en canonical schema worden hieronder kort toegelicht.

Canoniek data model

Dit patroon beschrijft een integratieoplossing waarbij voor interactie tussen twee of meer applicaties een gezamenlijk gestandaardiseerd datamodel geïntroduceerd wordt. Hierbij wordt het specifieke datamodel binnen de applicatie omgezet naar het gezamenlijke model. Gevolg hiervan is dan ook dat bij een integratie implementatie twee keer in plaats van eenmalig en conversie van datamodel plaats moet vinden. Dit is dan ook een veelgenoemd kritiekpunt voor dit patroon. Echter, zodra er meerdere applicaties behoefte hebben aan de data-entiteiten binnen deze implementatie, dan wordt dit patroon al snel interessant vanuit het oogpunt van efficiëntie.
In de uitwerking wordt ingegaan op de complexiteit van het introduceren van een canoniek datamodel. De kans op succes verkleint met name als gestreefd wordt naar een complete uitwerking van het datamodel voor een organisatie. Een stapsgewijze introductie waarbij de entiteiten uitgewerkt worden binnen het kader van integratieprojecten verhoogt de kans op succes.
Onderstaande afbeelding is een weergave van het patroon op basis van een integratie met twee bron- en twee doelsystemen.

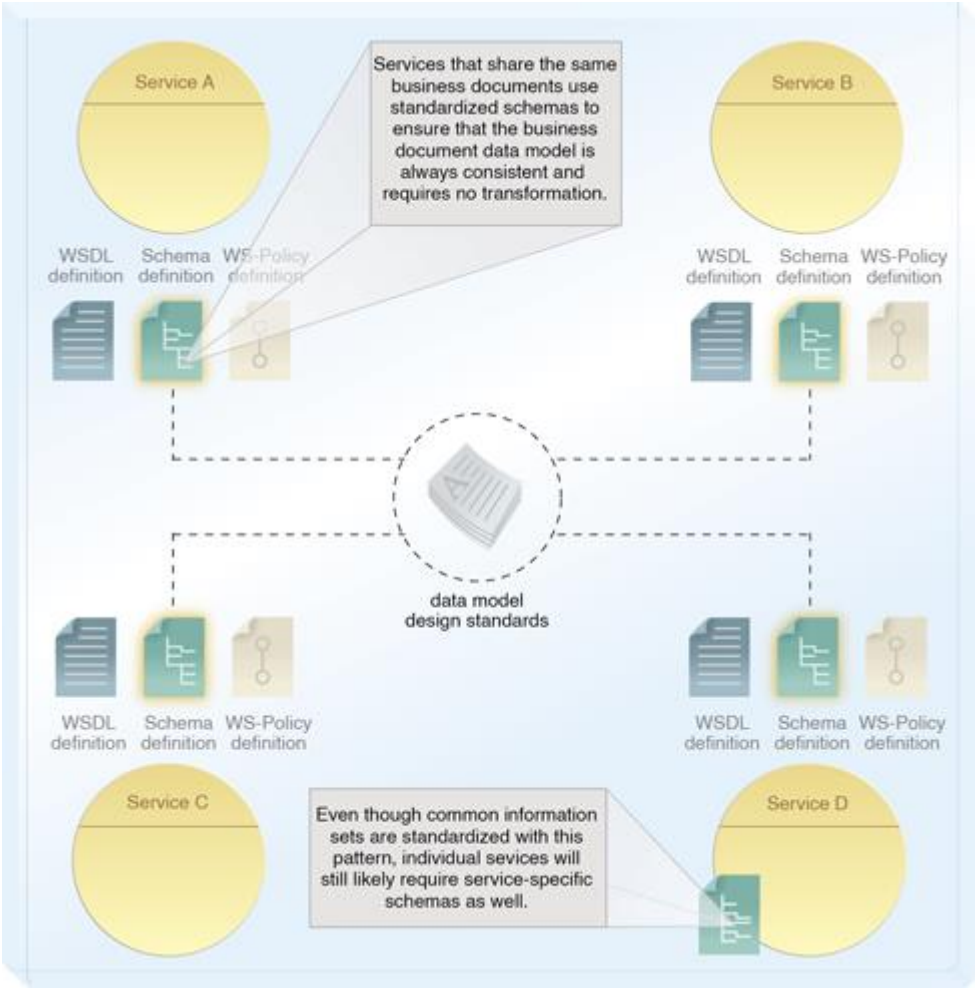


Bron: [Hohpe]

Canoniek schema

Dit patroon is afkomstig uit de servicegeoriënteerde architectuur en richt zich dan ook op op een aantal implementatieaspecten van service-oriëntatie. Het patroon is gericht op het uitwerken van een datamodel binnen de SOA-implementatie en gaat ervan uit dat het datamodel reeds aanwezig is.

Echter, in de praktijk maak ik regelmatig mee dat het canonieke datamodel nog niet aanwezig is en dus gelijktijdig ontwikkeld wordt met de inrichting van de service-architectuur. Zeker bij generieke gegevensverzamelingen zoals relatiegegevens, producten en ondersteunende sets zal deze werkwijze in meerdere iteraties verfijnd worden. Bij het inrichten van de schema's dient hier rekening mee gehouden te worden.
Daarnaast zie je organisaties die aangeven dat het introduceren van een Enterprise Service Bus een oplossingsrichting is voor het introduceren van een SOA. Echter, het introduceren van een ESB zonder na te denken over een canoniek model loopt grote kans te mislukken. Een ESB is namelijk een technische uitwerking van een probleem rond de generieke en herbruikbare datadefinities dat op bedrijfsniveau opgelost dient te worden voordat het geïmplementeerd wordt in een technische infrastructuur.
In onderstaande afbeelding is te zien dat het model grote overeenkomst heeft met het patroon van Hohpe, maar dat er een aantal specifieke implementatieaspecten zijn toegevoegd, zoals de XSD's, WSDL en Policy implementatie binnen de koppelvlakken.



Bron: [Erl]

Besluitvorming

In het hoofdstuk canonieke data-architectuur is reeds beschreven dat de data-architect de besluitvorming rond het canonieke model dient te ondersteunen. In dit hoofdstuk gaan we in op wie ondersteund dient te worden en op welke wijze deze besluitvorming tot stand komt. Het besluitvormingsproces heeft een belangrijke organisatie politieke dimensie. Bij de uitwerking van dit hoofdstuk wordt de besluitvorming daarom ingedeeld in een aantal politieke inrichtingsvormen.

Eigenaarschap

Bij de besluitvorming rond datadefinities binnen een canonieke architectuur zijn eigenaarschap of rentmeesterschap belangrijke aspecten. Het eigenaarschap bepaalt wie uiteindelijk de beslissingen neemt over de uitwerking, het gebruik en de implementatie van de (generieke) data-entiteiten en draagt er zorg voor dat deze entiteiten voldoende (data)kwaliteit hebben.
Rond het eigenaarschap van gegevens zijn er een aantal tendensen:

- **Niemand is eigenaar**, zeker bij gegevens met een hoge genericiteit is een eigenaar moeilijk aan te wijzen. Denk hierbij aan categorieën voor het indelen van de organisatie, producten en processen. Vaak zijn deze gegevens in meerdere applicaties opgenomen, wat de keuze van de juiste eigenaar bemoeilijkt**.**
- **Meerdere eigenaren,** de gegevensentiteiten worden in deze situatie door meerdere stakeholders gebruikt en iedereen voelt zich er in meer of mindere mate verantwoordelijk

voor. Als deze situatie gehandhaafd blijft, wordt het introduceren van een canoniek model bemoeilijkt doordat er een politieke strijd ontstaat.
- **Eigenaar binnen de ICT-organisatie**, omdat de ICT-organisatie ook zorg draagt voor het beheer van de gegevens lijkt het aanwijzen van een eigenaar binnen de ICT-afdeling een goed idee. Echter is dat veelal niet het geval, voor goed eigenaarschap is betrokkenheid en kennis nodig van de primaire processen in de organisatie. Die kennis is binnen een ICT-afdeling onvoldoende.

Samenvattend dient het eigenaarschap bij voorkeur belegd te worden bij één eigenaar binnen de primaire organisatie. Daarnaast dienen een aantal besluitvormings- en beheerprocessen ingericht te worden waarbij de eigenaar van de gegevens voldoende interactie heeft met de andere betrokkenen op basis waarvan de juiste beslissingen genomen kunnen worden. In onderstaande paragrafen werken we een aantal vormen uit ingedeeld naar een politieke metafoor.

Dictatuur

De besluitvorming rond het canonieke model kan plaatsvinden binnen een dictatoriaal besluitvormingsproces. In deze situatie heeft de eigenaar van het model dusdanig veel macht, dat er zonder veel inspraak met andere stakeholders bepaald kan worden hoe het model eruit ziet. Communicatie en documentatie om de stakeholders te informeren is veelal wel aanwezig om zorg te dragen voor een goede introductie van het canonieke model.
Deze vorm kan toegepast worden in situaties waar overeenstemming rond het model niet noodzakelijk is, bijvoorbeeld bij het uitvoeren van wetgeving of in situaties waarbij stakeholders voldoende andere prikkels hebben om in te stemmen met het model zonder dat men inspraak heeft. Voorbeelden zijn interfaces bij uitvoerende overheidsinstanties zoals de Belastingdienst of agentschappen.
Het/een voordeel van deze vorm is dat het opstellen van het model relatief eenvoudig is, omdat er met weinig stakeholders overeenstemming hoeft te worden bereikt. Nadeel van deze vorm is dat er veel effort/moeite? gestoken moet worden in informatieoverdracht. Daarnaast kan het datamodel dusdanig ver van de andere stakeholders afstaat dat de acceptatiegraad van het model laag is, omdat aansluiten op dit model teveel effort vraagt.

Democratie

In tegenstelling tot de dictatuur is de democratische besluitvorming gebaseerd op overeenstemming met alle stakeholders. Dat houdt in dat het canonieke model in samenspraak wordt uitgewerkt en geïntroduceerd. Tijdens het opstellen van het model heeft iedere betrokkene evenveel inspraak in het proces.
Deze vorm wordt toegepast in situaties waar het modelleren onderdeel is van een denk- of ontwikkelproces. Doordat iedere deelnemer evenveel inspraak heeft, ontstaat er commitment rond de opbouw van het model, de discussies vinden tenslotte plaats tijdens het opstellen van het model. Deze werkwijze wordt veelal toegepast in situaties waar alle stakeholders in machtsverhouding gelijkwaardig zijn. Denk bijvoorbeeld aan ketenintegratietrajecten binnen de overheid.
Het/een voordeel van deze vorm is dat er overeenstemming ontstaat over de opzet van het model. Bij introductie van het vastgestelde model zal er weinig weerstand zijn bij de betrokkene. Een nadeel van deze vorm is dat het besluitvormingsproces kan verzanden in een eindeloze discussie, onnodig lang duurt of dat er een consensus model ontstaat dat niemands problemen oplost.

Oligarchie

Oligarchie is een besluitvorming die dicht bij de dictatuur staat, echter bepaalt, in plaats van één stakeholder, een beperkte groep hoe het canonieke model eruit ziet. Naast deze groep is er een grote groep van stakeholders die belangen heeft bij de uitwerking van het model, echter hebben zij geen of nauwelijks invloed op het besluitvormingsproces.
Een voorbeeld van deze vorm is te vinden bij het opstellen van internationale integratiestandaarden. Denk hierbij aan ISO-, UN-CEFACT- en OASIS-standaarden. Hierbij zijn veelal een aantal grote (leveranciers)organisaties betrokken die de uitwerking van het model door kennis, invloed en onderhandelingsruimte bepalen
Een voordeel van deze vorm is dat de oligarchen grote invloed hebben en een groot deel van de stakeholders vertegenwoordigen. Daarnaast, zeker bij een leveranciersrol van de oligarchen, zullen zij zorg dragen voor een introductie van het model in de verschillende producten waarbinnen het canonieke model wordt ingezet. Een nadeel van deze vorm is dat het opgestelde model alleen aansluit bij de behoeften van de oligarchen en dat dit model veelal aansluit bij de kortetermijnbelangen van de groep. Daarnaast zie je vaak dat er twee of meer kampen ontstaan rond het probleemgebied. Een voorbeeld is het ontstaan van OOXML en ODF bij documentformaatstandaarden.

Coalitie democratie

Deze besluitvorm staat dicht bij de democratische vorm heeft maar een aantal beperkingen binnen de democratische vorm rond de discussies ondervangt. Er wordt namelijk gezocht naar een werkwijze waarbij het canonieke model wordt opgedeeld in onderdelen, waarna voor ieder onderdeel een verantwoordelijke wordt gekozen die dat onderdeel uitwerkt. Vervolgens worden de onderdelen samengevoegd en wordt het gehele model met alle betrokkenen besproken en eventueel bijgesteld.
Een voorbeeld van deze vorm komt uit de overheid bij het opstellen van het keteninformatiemodel Cites. Binnen dit programma zijn een vijftal overheidsorganisaties op basis van een logische indeling van het model gekomen tot een gezamenlijk model waarbij de indeling gebaseerd was op de domeineigenaren van de verschillende bronsystemen.
Een voordeel van deze vorm is dat er eenvoudig gezamenlijk overeenstemming wordt bereikt rond het gehele model, maar dat er geen discussies rond de detaillering plaatsvinden omdat de verantwoording hiervan bij één of enkele betrokkenen gelegd wordt voor de verschillende onderdelen. Hierin zit ook het grootste nadeel van deze opzet, de werkwijze is namelijk gebaseerd op wederzijds vertrouwen. Als dit ontbreekt wordt realisatie van een gezamenlijk model nauwelijks realiseerbaar.

Besluitvorm selecteren

Beter: Het kiezen van de besluitvormingswijze kan in bepaalde situaties lastig te bepalen zijn. Veelal heeft een data-architect daarin een beperkte invloed, met name omdat dit een politiek aspect is van de besluitvorming. Echter heeft de data-architect kennis rond de kansen van de werkwijze een adviserende rol naar de diverse politieke stakeholders. Bij advisering kan het onderstaande kwadrant behulpzaam zijn. Dit kwadrant is ingedeeld op basis van de machtsverhoudingen en de overeenstemmingsgraad.

		Overeenstemmingsgraad	
		Laag	**Hoog**
Machtsverhouding	**Overleg**	Democratie	Coalitie
	Macht	Dictatuur	Oligarchie

In het voorgaande hoofdstuk is ingegaan op de besluitvorming rond canonieke modellen. In de volgende paragrafen gaan we nader in op een aantal verschillende verschijningsvormen van canonieke modellen.

Berichtenboeken

De meest eenvoudige verschijningsvorm van een canoniek model is terug te vinden in bijvoorbeeld een berichtenboek of koppelvlakdefinitie. Het beschrijft de verschillende entiteiten in een tekstopbouw. In sommige gevallen wordt gebruik gemaakt van een tabelsgewijze opsomming van objecten, attributen en relaties.
In de afbeelding is een voorbeeld te zien van een willekeurig berichtenboek. Het laat zien dat de indeling van een berichtenboek vrij te kiezen is en dat er aan de notatievorm geen eisen worden gesteld.

3.1.4 *Chipper*

Chipper is de persoon die de chip bij het dier heeft ingebracht, bijvoorbeeld een dierenarts. Op dit moment kan alleen worden uitgegaan van een NatuurlijkPersoon. Bij de berichtverwerking geldt dat de gegevens van een natuurlijk persoon zijn opgenomen in de melding.

3.1.5 *Houder*

De eigenaar, houder of hoeder van het dier (overgenomen uit artikel 4c uit PDOC01-#208402-v1-Amvb Besluit identificatie en registratie van honden, versie Ministerraad).

Attributen

Naam	Card.	Type	Opmerkingen
klantnummer	[0..1]	klantnummerType	Het nummer dat een AANLEVERAAR (ADB) aan een klant heeft toegekend.
registratieNummerRechtspersoon		int	Registratienummer (ubn-nummer) dat door de Gezondheidsdienst voor Dieren aan de houder rechtspersoon is toegekend o.b.v. het Honden- en kattenbesluit 1999).

3.1.6 *raadpleegMeldingenAntwoord*

Attributen

Naam	Card.	Type	Opmerkingen
aantalMeldingen	[0..1]	maxMeldingenType	Aantal van het aantal meldingen dat gevonden is op basis van de bij raadpleegMeldingenverzoek ingevoerde gegevens.

Modellering op basis van een geschreven vorm zoals een berichtenboek wordt meestal toegepast in situaties waarin het kennisniveau van de stakeholders rond notatiewijzen laag is en waar kan worden volstaan met een niet gestandaardiseerde uitwerking van het model.
Het uitwerken van een canoniek model op basis van een berichtenboek kent de volgende voor- en nadelen:

Voordelen

- Het model is eenvoudig van opzet en niet gebaseerd op een notatiewijze

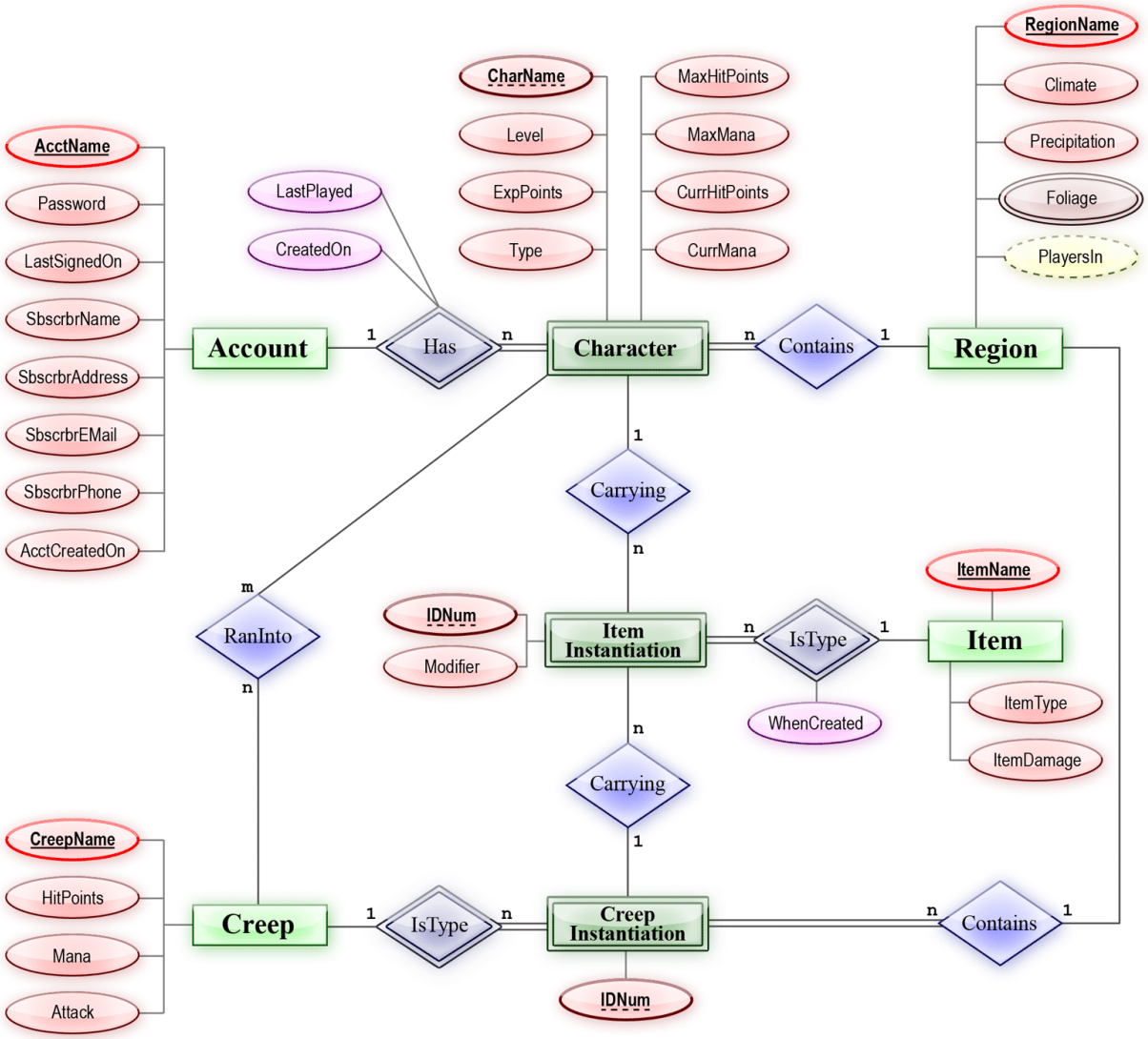
- De uitwerking is door iedere stakeholder te begrijpen zonder nadere toelichting op de notatiewijze
- Het model is niet gestandaardiseerd en daarom vrij in te richten afhankelijk van de situatie

Nadelen

- Door de vrijheid van uitwerking is deze notatievorm niet gestandaardiseerd
- Het document wordt snel omvangrijk en onoverzichtelijk
- Er kunnen interpretatieverschillen ontstaan tussen de opsteller en de lezer als er impliciete regels zijn geïmplementeerd
- Het is lastig om een document tijdens de beheerfase actueel te houden zonder inzet van tooling

ER-notatie

Vanuit het werkveld gegevensmodellering komt de Entiteit Relatie notatie. Binnen deze notatie kunnen entiteiten, attributen en de relaties tussen entiteiten gemodelleerd worden. Daarnaast kunnen een aantal extra aspecten, zoals cardinaliteiten van attributen en relaties, in de notatiewijze worden uitgewerkt.
In de afbeelding is een voorbeeld te zien waarin de entiteiten, attributen en relaties worden weergegeven.



Bron: Wikipedia

Modellering op basis van de ER-notatie wordt meestal toegepast in situaties waarin het kennisniveau van de stakeholders rond notatiewijzen van data hoog is en waar de stakeholders een technische achtergrond hebben.
Het uitwerken van een canoniek model op basis van de ER-notatie kent de volgende voor- en nadelen:

Voordelen:

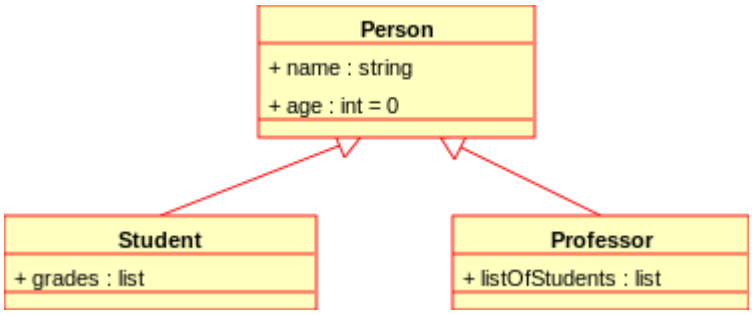
- Een breed toegepaste notatiewijze (zowel nationaal en internationaal)
- Er zijn vele vormen van tooling aanwezig (bijvoorbeeld powerdesigner, ERWin, Visio etc)
- Een consistente modellering en notatie voor overzicht en detail bij conceptuele, logische en technische ontwerpen

Nadelen:

- Het is technisch van opzet en vraagt daarom kennis en vaardigheden van de stakeholders
- Het modelleren van relaties van data-objecten met services, applicaties en processen is niet mogelijk
- Er is geen notatie aanwezig voor concepten als overerving, specialisatie en aggregatie

UML-klassendiagrammen

Vanuit de objectoriëntatie komt de UML-klassendiagram notatie/notatie van de UML-klassendiagram?. Deze notatie biedt de mogelijkheid om gegevensstructuren op basis van een uitgebreidere notatie dan bijvoorbeeld de ER-notatie te modelleren. De notatie is bijzonder krachtig en kent een notatiewijze voor zowel entiteiten, attributen als relaties. Daarnaast kunnen entiteiten als attribuut binnen een entiteit opgenomen worden en kan van verschillende onderdelen de zichtbaarheid aangegeven worden.
De onderstaande afbeelding is een eenvoudig voorbeeld van een UML-klassendiagram.



Bron: Wikipedia

De UML-klassendiagram wordt voornamelijk toegepast in situaties waar de stakeholders een achtergrond in de ICT hebben en bekend zijn met de concepten die toegepast worden binnen objectoriëntatie en softwareontwikkeling en -modellering.

Voordelen:

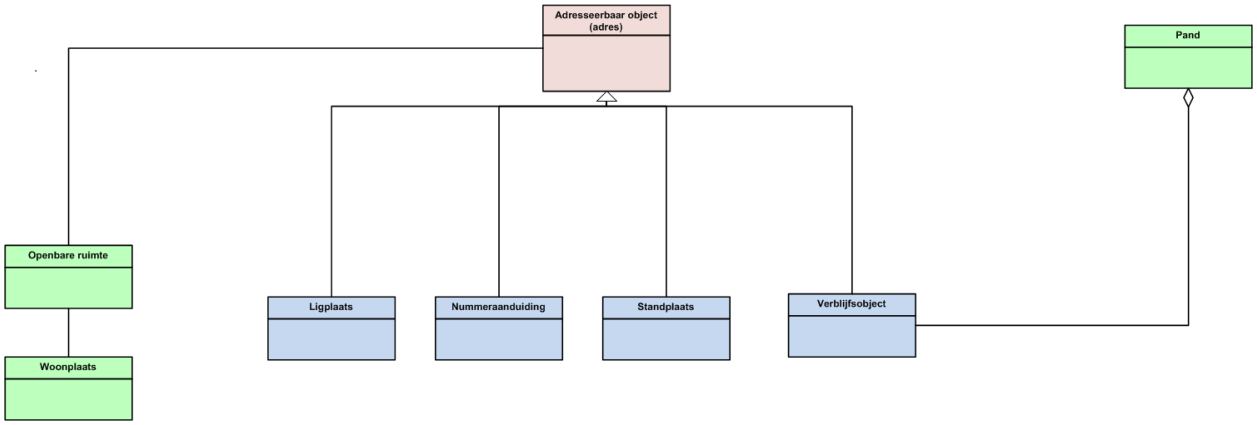
- Een internationale en breed geaccepteerde standaard
- Interactie met leveranciers wordt eenvoudiger
- Het genereren van diverse andere output zoals XSD's maar ook berichtenboeken

Nadelen:

- Notatie is rijk en modellen worden snel complex
- Kans op te grote mate van detaillering
- Het uitwerken van de richtlijnen voor viewpoints is noodzakelijk
- Het modelleren van relaties met services en processen is niet mogelijk in modellering

ArchiMate

ArchiMate is een open architectuurtaal waarmee het mogelijk is om een organisatie vanuit IT perspectief te modelleren. De taal heeft grote overeenkomsten met bijvoorbeeld UML, maar richt zich daarbij duidelijk op een hoger abstractieniveau.
De notatiewijze van ArchiMate is zeer uitgebreid en kent een duidelijke indeling van concepten op meerdere lagen in een architectuur. Naast de uitgebreide set van concepten is ArchiMate ingedeeld in verschillende “viewpoints”. Dit zijn uitgewerkte weergaven voor verschillende stakeholders. Daarnaast is in ArchiMate beschreven hoe je extensie kunt implementeren in situaties waar de taal onvoldoende detail biedt.



Voordelen

- Een internationale standaard
- Andere concepten zoals service, applicatie en proces maken deel uit van notatie
- Objectmodellering is mogelijk
- Gelaagdheid door business, data en artifact
- Extensie in de taal is mogelijk
- Gebruik van viewpoints
- Tooling is aanwezig (ook als open source software)

Nadelen

- Attributen van data objecten zijn niet standaard aanwezig
- Diagrammen worden snel complex en technisch
- Niet geschikt voor alle stakeholders vanwege complexiteit in de notatiewijzen

Hybride vormen

Bij het uitwerken van canonieke data-architectuurmodellen neemt de complexiteit door omvang van het model snel toe. Zeker in situaties waarbij verschillende stakeholders op verschillende wijzen betrokken zijn bij het uitwerken van het model. Denk hierbij bijvoorbeeld aan beslissers rond de opzet van het model, maar ook aan ontwikkelaars die het model gaan implementeren in bijvoorbeeld componenten en (web)services.

In bovenstaande situaties zal één enkele notatiewijze veelal onvoldoende invulling geven aan de behoeften van meerdere stakeholders. Er zal in die situatie behoefte ontstaan aan een hybride vorm van notatiewijzen. Vaak zal deze notatiewijze een combinatie zijn van een aantal van bovenstaande notatiewijzen. Een aantal toepassingen zijn:

- ArchiMate en UML-klassendiagram
- ArchiMate en extensie voor attribuutmodellering
- ER en berichtenboek
- UML en berichtenboek

Implementeren

In de voorgaande paragraaf is beschreven op welke wijze een canonieke data-architectuur gemodelleerd kan worden. In deze paragraaf gaan we hier verder op in door aan te geven op welke wijze een canoniek model kan bijdragen aan verschillende aspecten van (data-)architectuur.

Applicatie-integratie

Applicatie- of systeemintegratie is een deelgebied van de ICT dat evolutionair ontstaan is. In het begin werden vrijwel onzichtbaar afspraken gemaakt tussen gebruikers en beheerders om gegevens op enigerlei wijze met elkaar uit te wisselen. De voordelen van applicatie-integratie waren al in een vroeg stadium evident. Gegevens die door andere beheerd worden en een hogere kwaliteit hebben dan als men ze zelf moet beheren zijn waardevol en worden daarom uitgewisseld.
Echter, door de evolutionaire ontwikkeling ontstonden op een gegeven moment problemen bij het beheer van de verschillende koppelingen. Zeker bij het beheer van een register met veelgebruikte gegevens zie je een wildgroei van koppelingen ontstaan.
Het gevolg is dat men begint met het in kaart brengen van de koppelingen die aanwezig zijn en de koppelingen die gewenst zijn. Veelal wordt in deze situatie ook begonnen met het beschrijven van de gegevensverzamelingen en indien mogelijk wordt een vorm van standaardisatie toegepast.
Zeker in omvangrijke organisaties of in situaties waar een bronregister een groot aantal afnemers kent en de diversiteit van de verschillende afnemers groot is biedt canonieke datamodellering goede mogelijkheden om complexiteit te reduceren of beheersbaar te maken. Bijvoorbeeld het modelleren van de verschillende entiteiten uitgewerkt in de vorm van een koppelingenregister.
Afhankelijk van de scope van de canonieke datamodellering kunnen verschillende vormen van modelleren worden toegepast. Ligt de scope met name op de inrichting van de data-entiteiten binnen de koppelingen, dan zijn UML-klassendiagrammen of ER-diagrammen goed toepasbaar. Wil men ook in kaart kunnen brengen wat de bron- en doelinformatiesystemen zijn, dan is ArchiMate goed inzetbaar.

Service-oriëntatie

Service-oriëntatie is een architectuurvorm waarbij standaardisatie van de interface een belangrijk aspect is. Door deze standaardisatie ontstaat flexibiliteit voor de afnemende werkprocessen. Hierdoor wordt de organisatie in zijn geheel wendbaarder zonder dat de IT resources een beperkende factor zijn.
Deze standaardisatie vindt plaats op enerzijds de toegepaste protocollen en integratiewijze. Anderzijds wordt veelal gezocht naar een gestandaardiseerd datamodel binnen met name de entiteit service laag.
Dit gestandaardiseerde datamodel wordt gebruikt om transformatie tijdens het transport binnen de services zoveel mogelijk te voorkomen. Het wordt daarnaast gebruikt om het hergebruik van services met een gestandaardiseerde interface te stimuleren.

In het eerste hoofdstuk is reeds een patroon geschetst vanuit de service-oriëntatie, wat een goed voorbeeld is van een vorm van canonieke datamodellen binnen de schema's. Schema's zijn de beschrijving van (web)services die binnen een SOA ingezet worden om het datamodel te definiëren. Het ontsluiten van de datamodellen binnen deze schema's, bijvoorbeeld in de vorm van een serviceregister is een veel toegepaste vorm om een canoniek datamodel te ontsluiten voor doelen met betrekking tot beheer, documentatie en hergebruik.
Canonieke modellen binnen schema's en serviceregisters worden veelal beschreven op basis van UML-klassendiagrammen en een verbijzondering hiervan. Reden is dat modelleertooling veelal in staat is om op basis van deze modellen documentatie en technische uitwerkingen van deze modellen, bijvoorbeeld in de vorm van XSD-bestanden, te genereren.

Enterprise- of referentie-architectuur

Bij het uitwerken van een enterprise-architectuur wordt vaak gekozen voor het beschrijven van de organisatie en de ICT van de huidige en toekomstige situatie. De verschillende tussensituaties worden bij een omvangrijke verandering van organisatie of ICT eveneens beschreven. Een/het onderdeel van de verschillende beschrijvingen omvat in veel gevallen ook aspecten van data- en bedrijfsobjecten.
In een aantal sectoren zie je dat er naast werkprocessen een duidelijk overeenkomst bestaat in data- of bedrijfsobjecten. Door het gezamenlijke aspect van de verschillende elementen te beschrijven in een referentie-architectuur ontstaat een vorm van standaardisatie op verschillende niveaus van de betrokken organisaties. Zo ontstaan er bijvoorbeeld standaardbenamingen voor processen. Ook zie je in die situatie vocabulaires ontstaan die standaarddefinities van entiteiten en de bijbehorende attributen omvatten.
In beide architectuurvormen kan een canoniek model een goede bijdrage leveren aan de toegankelijkheid van de data-entiteiten en daarmee de complexiteit reduceren, zeker in het geval van een gelaagde uitwerking van de entiteiten.
Het opstellen van canonieke modellen in het kader van een enterprise- of referentie-architectuur zal veelal op basis van ArchiMate uitgewerkt worden. Bij een gelaagde uitwerking zal gezocht worden naar een hybride uitwerking met ArchiMate voor de bedrijfslaag en UML-klassendiagrammen voor de informatielaag.

Standaardisatie

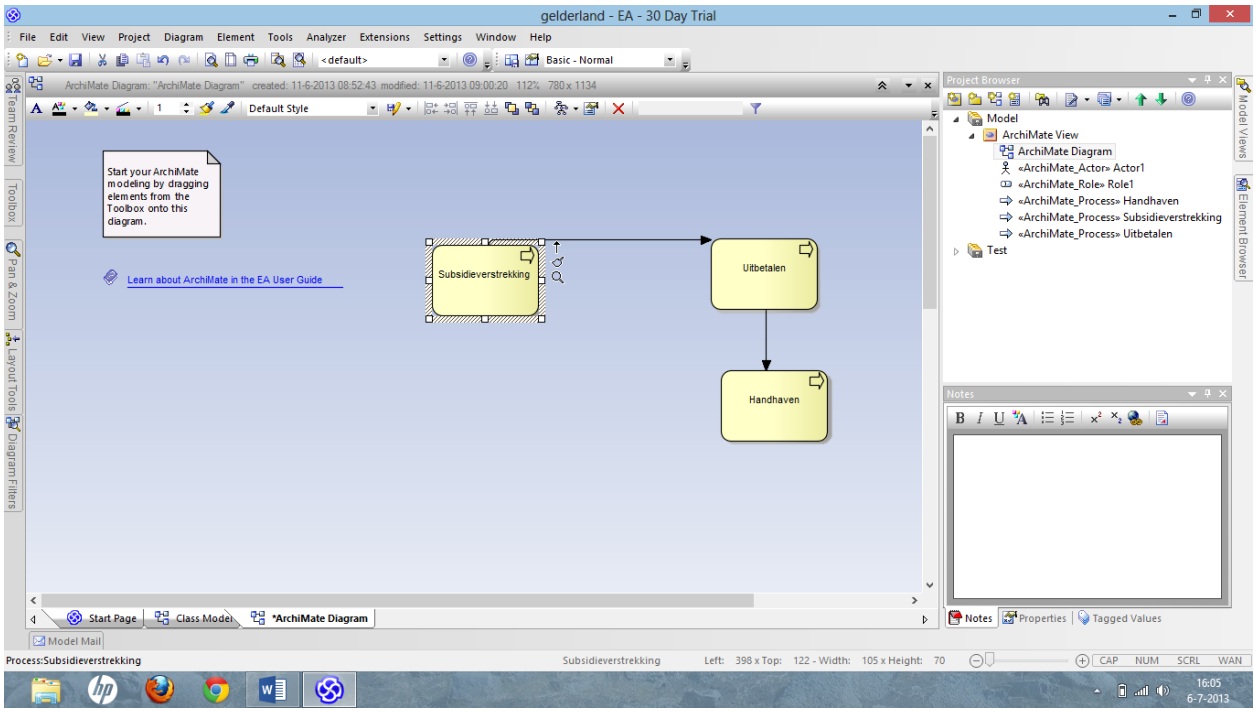
Met standaarden hebben veel mensen een haat-liefdeverhouding. Zo ben ik elke dag blij dat men in het verleden de stopcontacten gestandaardiseerd heeft. Echter, bij het inrichten van ICT kunnen standaarden lastig te implementeren zijn. Denk aan open standaarden als UBL of het internationale datamodel van bijvoorbeeld de douane. Daarnaast zie je dat standaarden in een aantal gevallen achter de technologische mogelijkheden aan lopen (mobile).
Neemt niet weg dat het belang van (open) standaarden in het kader van interoperabiliteit evident zijn. Bij het opstellen van een (open) standaard op het vlak van bedrijfs- of data-objecten zal een canoniek model noodzakelijk zijn.
Bij het uitwerken van een open standaard zal als canonieke modelleerwijze veelal gekozen worden voor een beschrijvende vorm, bijvoorbeeld in de vorm van berichtenboeken. Ook wordt vaak gekozen voor een hybride vorm met berichtenboeken en bijvoorbeeld UML-klassendiagrammen.

Tooling voor Canonieke data-architectuur

Bij het uitwerken van een data-architectuur en het opstellen van een canoniek model neemt de complexiteit van het model snel toe. Zeker bij grote organisaties of een omvangrijk deelgebied.

Complexiteit enerzijds door de hoeveelheid elementen, anderzijds door de diversiteit van deze elementen en als laatste door de vele onderlinge relaties.
In die situatie kan tooling behulpzaam zijn bij het opstellen van een canoniek data model. Deze tooling richt zich meestal op meerdere aspecten van canonieke datamodellering. Denk hierbij aan:

- Registeropbouw van bijvoorbeeld data-objecten, hun attributen en onderlinge relaties.
- Documenteren van deze data objecten
- Opstellen van diagrammen om (data-)objecten, hun attributen en onderlinge relaties inzichtelijk te maken.
- Analyseren en valideren van canonieke datamodellen
- Genereren van diverse afgeleide weergaven zoals XSD-bestanden, tabelstructuren, documentatie, berichtenboeken
- Verschillende vormen van zoeken en navigeren ten behoeve van diverse stakeholders.



Tooling is verkrijgbaar in allerlei vormen. Van zeer eenvoudig tot geavanceerd en van goedkoop tot duur. Het gaat aan dit whitepaper voorbij om een lijst van de tools op te nemen. Momenteel wordt er gewerkt aan een digitale lijst van verschillende tools ten behoeve van data-architectuur. Deze lijst zal zodra gereed ontsloten worden via de architectuurassistent website.

Tot slot

Dit whitepaper is een inleidende beschrijving van canonieke data-architecturen. Het heeft een aantal aspecten van canonieke data-architectuur beschreven. Voor een aantal aspecten is een nadere detaillering noodzakelijk. Denk hierbij aan bijvoorbeeld een gedetailleerde uitwerking van de modelleerwijzen een overzicht van de verschillende tools.
Daarnaast zijn onderwerpen rond canonieke data-architectuur nog niet uitgewerkt die bij een nadere detaillering niet mogen ontbreken. Bijvoorbeeld datakwaliteiten in relatie tot entiteiten, beheer van canonieke data-architectuur en het inzetten van canonieke data-architectuur in andere deelgebieden van enterprise-architectuur. In volgende artikelen en whitepapers worden deze onderwerpen uitgewerkt en beschikbaar gesteld via www.architectuurassistent.nl.

Referenties

Erl, Thomas (2003). *SOA design patterns.* Prentice Hall.

Hohpe, George et al (2008). *Enterprise Integration Patterns : Designing, Building, and Deploying Messaging Solutions*, 2004, Addison Wesley.

z.a., z.j. *Wikipedia.org.*