Single Point of Definition Metadata

Een betrouwbare weg naar Metadata Management

Peter Alons (Dr. P.W.F. Alons)
Origin/Business Management Solutions (BMS)

In de jaren '90 stelde Michael Hammer, co-auteur van het boek "*Reengineering the Corporation*" [1], dat 85% van de pogingen om het bedrijf significant te verbeteren zullen falen. Ik waag het erop om hier op deze uitspraak te parafraseren: zelfs in deze tijd zullen vele ontwikkelprojecten voor grote database systemen, zoals Data Warehouses, tegen serieuze problemen aanlopen, die uiteindelijk tot falen zullen leiden. Nu zijn hiervoor diverse redenen aan te geven die beslist ook een serieuze bespreking waard zijn, maar in dit artikel niet aan bod komen. Misschien dat ik daar in de toekomst nog eens gelegenheid toe krijg. Meer specifiek voor dit artikel luidt de parafrase: zelfs in deze tijd zullen vele pogingen om adequaat metadata management op te zetten, zoals voor Data Warehouses, tegen serieuze problemen aanlopen, die uiteindelijk tot falen zullen leiden. En ook dan is de vraag natuurlijk: waarom?

De problemen liggen in de eerste plaats op het gebied van de ontwikkeling van informatiesystemen en in de tweede plaats op het gebied van metadata management zelf. Bij de ontwikkeling van informatiesystemen vloeien de problemen voort uit gebreken in de methodische aanpak en op het gebied van kennis en vaardigheden. Aldus gerubriceerd hebben we:

Gebreken in de methodische aanpak:

1. Onvoldoende gebruikersparticipatie;
2. Het gebruik van niet-conceptuele aanpakken;
3. Onvoldoende documentatie.

Het eerste gebrek leidt onherroepelijk tot slechte validatie van de informatiemodellen, het tweede tot zwakke ontwerpen, en het derde tot onderhoudsproblemen. Goede validatie is overigens niet alleen een zaak van voldoende gebruikersparticipatie. Hoe serieus we gebruikersparticipatie ook nemen (en wie doet dat niet tegenwoordig), we kunnen niet zonder werkelijk effectieve validatietechnieken. We moeten immers onomstotelijk kunnen vaststellen, dat de gebruikers echt akkoord zijn met onze modellen. Kortom, wat we nodig hebben is *af*doende gebruikersparticipatie.

Tekortkomingen in kennis en vaardigheden:

1. Onvoldoende kennis of gebruik van gedegen ontwikkelmethoden voor informatiesystemen;
2. Onvoldoende kennis of gebruik van moderne casetool-ondersteuning.

De eerste tekortkoming leidt doorgaans tot slechte systemen en de tweede tot onnodig tijdverlies bij het modelleren, ontwerpen, prototypen en ontwikkelen van applicaties.

Bij het metadata management zelf zijn de problemen gelegen in de locatie en de distributie van metadata. Aldus gerubriceerd hebben we:

Gebreken in de locatie van metadata:

1. De metadata zijn opgeslagen op verschillende, niet samenhangende locaties;
2. De metadata zijn gedeeltelijk opgeslagen in handmatig onderhouden documenten;
3. De metadata zijn onvoldoende georganiseerd.

De eerste twee gebreken leiden tot *update-anomalieën*: inconsistenties in de informatie (in dit geval voor de informatie over de bedrijfsinformatie). Vooral het tweede gebrek vereist een hoge mate van discipline om de metadata correct te onderhouden, en die wordt na verloop van tijd niet meer opgebracht. Het derde gebrek uit zich in onvoldoende afspraken over standaards, versiebeheer enzovoort.

Gebreken in de distributie van metadata:

1. De metadata worden gedistribueerd in ouderwetse ringmappen;
2. Er wordt geen of matig gebruik gemaakt van intranet- en web-faciliteiten;
3. Er wordt geen gebruik gemaakt van repository-transformaties.

Hier uit zich de eerste tekortkoming in het niet toereikend up-to-date houden van de gedistribueerde metadata. De tweede tekort-

koming leidt ertoe, dat eindgebruikers onvoldoende en ontoereikende toegang tot de metadata hebben. De derde tekortkoming leidt simpelweg tot volstrekt onnodig verlies van informatie in de communicatielijnen van metadata.

Mijn praktijkervaringen hebben mij geleerd, dat alle genoemde euvels nog heel geregeld om zich heen grijpen. Zo zeer zelfs, dat zij geleid hebben tot de verspreiding van een geheel nieuwe leer over metadata: "Vergeet het bijhouden van metadata maar gewoon. Niemand doet het correct en het is niet te betalen, dus waarom zouden we het überhaupt nog doen…". Deze leer is wellicht aantrekkelijk voor managers die op korte termijn graag geld besparen, maar het is natuurlijk niets meer dan de problemen uit de weg gaan en lijkt mij dan ook te kort door de bocht. In dit artikel wil ik een wat serieuzere kijk op deze problematiek voor het voetlicht brengen. De kern van de zaak daarbij is niet, *dat* we bovenstaande gebreken moeten vermijden. Het gaat erom, *hoe* we ze het beste vermijden. Daarvoor moet het op langere termijn betaalbaar zijn, en als het kan ook nog een beetje leuk om te doen. Dit wordt in hoge mate bepaald door de methoden en technieken die we gebruiken en dat is dan ook waar ik me in dit artikel op richt. Ik begin daarbij met de aanpak van systeemontwikkeling zelf en ga daarbij met name wat dieper in op een goede methode voor conceptuele informatie-modellering. Daarna bekijken we de consequenties daarvan voor het metadata management zelf.

Het ontwikkelen van informatiemodellen

In [2] wordt aangegeven, dat informatie-systemen vanuit drie invalshoeken kunnen worden beschouwd:

1. *Informatie:* Welke (soorten van) informatie zijn relevant voor de communicatie-processen die door het informatiesysteem ondersteund moeten worden. Anders gezegd: welke typen van feiten zijn belangrijk.
2. *Processen:* Welke invoer-, uitvoer-, onderhoud- en afleidingsprocessen moeten op de betreffende informatie worden toegepast. Anders gezegd: welke processen moet het informatiesysteem bieden en wat moeten ze precies doen?
3. *Gedrag:* Welke gebeurtenissen activeren bovengenoemde processen. Anders gezegd: wat zijn de *events* (zoals menukeuzen) die er voor zorgen dat de processen worden uitgevoerd. Vooral in real-time systemen is deze invalshoek belangrijk.

Aan deze drie invalshoeken kunnen we nog een vierde toevoegen:

4. *Presentatie:* Hoe kan de informatie het beste door het systeem gepresenteerd worden.

Welke van deze invalshoeken het belangrijkst zijn hangt af van het soort informatiesysteem. Bij bijvoorbeeld processturende en proces-controlerende real-time systemen zijn, zoals al aangegeven, vooral de processen en het gedrag belangrijk. Bij de informatiesystemen in *gegevensintensieve* toepassingsgebieden ligt de nadruk op de informatie en presentatie. Met gegevensintensief bedoelen we, dat de informatie die in de systemen verwerkt wordt kwantitatief omvangrijk is, zowel op instantie-niveau (veel verschillende feiten van dezelfde soort, zoals bijvoorbeeld weergegeven door de records in een verkooptabel) als op typeniveau (veel verschillende soorten van feiten, bijvoorbeeld verkopen, verzendingen, voor-raden, klantlocaties enz.). In deze gegevens-intensieve toepassingsgebieden is de aard van de relevante informatie (die zich uit in de gebruikte gegevensstructuren, zoals een 3NF-model of stermodel) stabieler dan de relevante processen en events. Bovendien kunnen de andere invalshoeken alleen goed worden gespecificeerd in termen van de te verwerken informatie. Verder kunnen gewenste wijzigingen in de processen en het gedrag met de moderne hulpmiddelen meestal veel eenvoudiger worden gerealiseerd dan wijzi-gingen in de gegevensstructuren. Natuurlijk is bij dit soort systemen ook de presentatie erg belangrijk. Voorbeelden hiervan zijn de grafische user-interfaces van de EIS- en OLAP-applicaties die de informatie in een Data Warehouse weergeven. Maar ook deze kunnen tegenwoordig in hoge mate met moderne casetools gegenereerd worden.

We kunnen in de invalshoek informatie een zinvolle structuur aanbrengen. We doen dit door vier niveaus van modellering te onderscheiden.

1. *Het Conceptuele niveau:*
   Conceptueel wil zeggen begripsmatig. Op dit niveau streven we ernaar een zo

volledig mogelijk *conceptueel informatiemodel* vast te leggen. De focus ligt hierbij geheel op het domeingebied: de werkelijkheidsbelevenis van de bedrijfsdeskundigen. Een vaak gebruikte term hiervoor is het UoD (Universe of Discourse). We zouden dit niveau met recht ook het *eindgebruikersniveau* kunnen noemen. Het gemaakte conceptuele informatiemodel dient vrij te blijven van redundante informatie: net zoals we bij een relationeel ontwerp ernaar streven om de informatie in de database redundantievrij op te slaan, moet dit ook gebeuren met de informatie over de informatie, de metadata dus.

2. *Het Logische niveau:*
   Hierin ligt de focus geheel op het gegevensmodel, d.w.z. de structuur en de samenhang van en de relaties tussen de verschillende gegevenselementen in het UoD. Dit zou ik ook het *ER-niveau* willen noemen (ER = Entities and Relationships). Voor dit niveau geldt net als op het conceptuele niveau, dat het model vrij hoort te zijn van redundante ontwerpinformatie. In de praktijk worden ER-diagrammen vaak vol redundante ontwerpinformatie gepresenteerd, waardoor ze een mengeling worden van relationele schema's en correcte ER-diagrammen. Zo worden bijvoorbeeld bij entiteittypen verwijzende attributen die het gevolg zijn van een relatie expliciet getoond. Zolang het ER-tool dit als afgeleide informatie genereert hoeft dat echter nog geen probleem te zijn.
3. *Het Fysieke niveau:*
   Op dit niveau ligt de focus op het fysieke platform, d.w.z. op de consequenties van het gekozen DBMS waarin het logische gegevensmodel wordt geïmplementeerd. Hierbij denken we nog niet aan allerlei niet-conceptuele toevoegingen aan het model, maar aan zaken als het opnemen van afleidbare ontwerpinformatie, zoals verwijzende sleutels, en noodzakelijke naamsveranderingen ten gevolge van beperkingen van het gekozen DBMS. Dit zouden we ook het *RM-niveau* kunnen noemen (RM = Relationele Model).
4. *Het Technische niveau:*
   Hierbij ligt de focus op het interne gebruik van het platform, d.w.z. op de consequenties van het interne gegevensmanagement van het gekozen DBMS. Hieronder vallen bijvoorbeeld de door de DBA nuttig geachte niet-conceptuele toevoegingen aan het model, zoals indexen en 'partition-keys'. Dit zouden we dus ook het *DBMS-niveau* kunnen noemen.

De eerste twee niveaus van deze indeling vormen samen de *ontwerplaag* van de gegevensmodellering, de laatste twee de *implementatielaag*. Het logische en het fysieke niveau duiden we tezamen aan met de *ER-laag.* En hiermee komen we meteen bij de crux van metadata management. Goed, beheersbaar, en daarmee verantwoord en betaalbaar metadata management vereist, dat de bovengenoemde niveaus goed en zorgvuldig gescheiden worden gehouden. Nu kan dit in het algemeen goed worden gedaan voor de laatste drie niveaus, omdat in deze niveaus goede ondersteuning kan worden gevonden. Voor het logische en fysieke niveau bestaan er bijvoorbeeld vele redelijke tot uitstekende ER-tools, terwijl voor het technische niveau vele goede 4GL-tools en RDBMS-en kunnen worden gevonden. Bovendien kunnen deze tools doorgaans goed computerondersteund met elkaar communiceren. Het probleem zit hem in het conceptuele niveau. Daarvoor bestaat in het algemeen slechts zwakke ondersteuning. Gelukkig gloort er licht aan de horizon: sinds een paar jaar zijn er enkele methoden die dit niveau behoorlijk ondersteunen. De beste daarvan is mijns inziens op dit moment FCO-IM.

FCO-IM staat voor *Volledig Communicatiegeoriënteerde Informatiemodellering* (Engels: *Fully Communication Oriented Information Modeling)*. Deze methode wordt volledig beschreven in [2] en aan de hand van dit boek onderwezen aan diverse hogescholen en universiteiten in Nederland. Inmiddels is dit boek in het engels en duits vertaald en heeft FCO-IM ook in de USA ingang gevonden. Zo heeft de atoomwapengigant Sandia National Laboratories zich formeel FCO-IM gestandaardiseerd verklaard. De methode wordt geheel conform de theorie ondersteund door de voortreffelijke FCO-IM Casetool van Ascaris. Het voert hier te ver deze methode in detail te beschrijven, daarvoor verwijs ik naar de genoemde literatuur en de website www.FCO-IM.net. Hieronder laat ik middels een concreet voorbeeld de belangrijkste principes van FCO-IM zien, omdat die relevant zijn voor het vervolg van dit verhaal.

Doelstellingen en claims van FCO-IM

FCO-IM beoogt volledige ondersteuning te geven bij het opstellen en benutten van een conceptueel informatiemodel, het formele resultaat van een informatieanalyse. Dit conceptuele informatiemodel wordt afgeleid op basis van concrete en relevante gebruikers-voorbeelden. Deze voorbeelden worden in samenspraak met de domeindeskundigen verwoord in door hen volledig begrijpelijk en acceptabel geachte zinnen. Deze zinnen worden elementair gemaakt en daarna geclassificeerd, gekwalificeerd en geanalyseerd. Hierdoor wordt een model gemaakt dat bestaat uit een verzameling relevante feittypen, vastgelegd via hun *harde* semantiek die de drijver is van de structuur en integriteitregels van deze feit-typen, en hun *zachte* semantiek, de feittype-formuleringen, die de precieze betekenis van de feittypen voor de domeindeskundigen weergeven. Om beter te begrijpen wat hiermee wordt bedoeld is een voorbeeld op zijn plaats.

Stel dat een domeindeskundige ons de tabel Aircraft laat zien zoals gegeven in tabel 1:

Aircraft

Regcode	Naam	Aflever datum	Subtype	Subtype Beschr.	Type	Type Beschr.	Eigenaar
PH-BDA	Willem Barentsz	09-1986	733	Boeing 737-300	737	Boeing 737	KL
PH-BXB	Falcon	05-1999	738	Boeing 737-800	737	Boeing 737	KL
PH-BFN	Nairobi	04-1993	744	Boeing 747-400	747	Boeing 747	KL

Tabel 1: Informatie over vliegtuigen

Om de reeks gegevenselementen in elk record te begrijpen moeten we ons realiseren, dat elk record een verhaal vertelt. De domein-deskundigen zegt dat het eerste record vertelt, dat de KLM een Boeing 737 bezit met registratiecode PH-BDA, genaamd de Willem Barentz, die in september 1986 aan de KLM werd afgeleverd, en een Boeing 737-300 is, d.w.z. type 737 en subtype 733. De andere records kunnen nu overeenkomstig gelezen worden. We knippen deze zin op in *elementaire* zinnen, d.w.z. zinnen die maar één eenduidig feit weergeven. Dan ontstaat de verzameling zinnen in tabel 2:

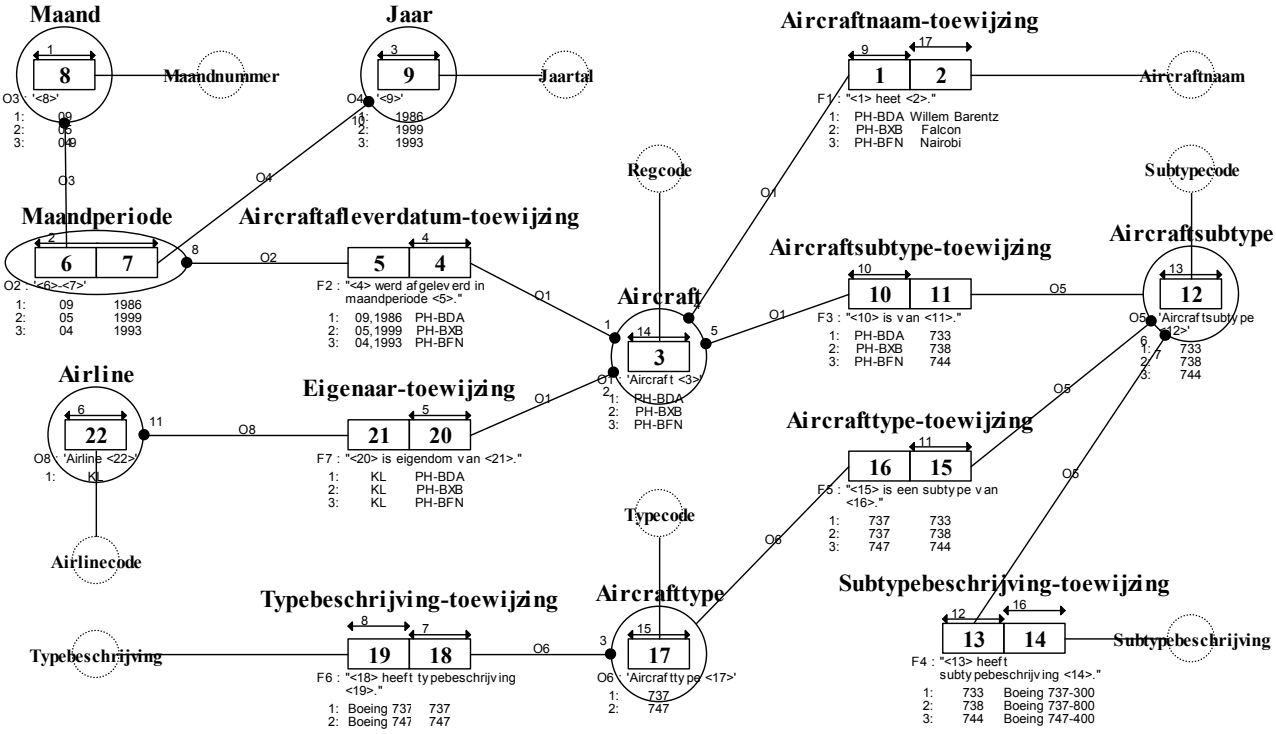
Zin (feitexpressie)	Classificatie en Kwalificatie (feittype)
Aircraft PH-BDA heet Willem Barentz	Aircraftnaam-toewijzing
Aircraft PH-BDA werd afgeleverd in maandperiode 09-1986	Aircraftafleverdatum-toewijzing
Aircraft PH-BDA is van Aircraftsubtype 733	Aircraftsubtype-toewijzing
Aircraftsubtype 733 heeft subtypebeschrijving Boeing 737-300	Subtypebeschrijving-toewijzing
Aircraftsubtype 733 is een subtype van Aircrafttype 737	Aircrafttype-toewijzing
Aircrafttype 737 heeft typebeschrijving Boeing 737	Typebeschrijving-toewijzing
Aircraft PH-BDA is eigendom van Airline KL	Eigenaar-toewijzing

Tabel 2: Zinnen en feittypen

Elke zin in deze tabel vertegenwoordigt precies één feittype: voor elk feittype kunnen we uit de drie records in tabel 1 drie van dit soort zinnen maken. Dit is classificeren: het bij elkaar zetten van gelijksoortige zinnen behorend bij één feittype. Kwalificeren houdt in dat aan elk van de feittypen een eigen naam wordt gegeven, zoals gedaan in kolom 2 van tabel 2. Nu komen in elk zintype zindelen voor die per zin kunnen veranderen. Ook die krijgen een eigen naam. Dit is het voortzetten van het classificeren en kwalificeren. Zo verwijzen de zinnen van de eerste drie feittypen in tabel 2 alledrie naar een object Aircraft PH-BDA. Dit object correspondeert ook met een uniek feit, weer te geven door een feitformulering als: "Er bestaat een Aircraft aangeduid met PH-BDA" (maar we hebben dat zintype niet echt nodig). Laten we dit feittype Aircraft noemen (tabel 1 bevat drie instanties van dit feittype). Het zinsdeel Aircraft PH-BDA heeft ook weer een variabel deel, PH-BDA, dat volgens onze domeindeskundige een Regcode is. Omdat dit objecttype niet meer dan een datadomein is, dat alle toegestane aircraft-identificaties bevat, heet zo'n objecttype een labeltype. Zo levert het kwalificatieproces nog de feittypen Aircraft-subtype, Aircrafttype, Maandperiode, Maand, Jaar en Airline, en de labeltypen Aircraftnaam, Maandnummer, Jaartal, Subtypecode, Sub-typebeschrijving, Typecode, Typebeschrijving en Airlinecode. De domeindeskundige is het

met al onze zinnen en namen eens en vertelt vervolgens op verzoek, dat elk vliegtuig maar één Regcode en naam kan en moet hebben. Ook kan en moet het maar van één subtype zijn, dat op zich weer tot één type behoort. Tenslotte heeft het vliegtuig maar één Airline als eigenaar. Deze *analyse* van de feittypen levert aldus per feittype op wat daarin uniek is, en wat verplicht. Al deze dingen bij elkaar - de resultaten van onze classificatie, kwalificatie en analyse - noemen we de harde semantiek van de betreffende feittypen. Deze drijven volledig hun structuur en de integriteitregels waaraan ze moeten voldoen. Alle zinnen bij de verschillende feittypen, die de betekenis van de feittypen voor de domeindeskundigen weergeven, noemen we de zachte semantiek van de feittypen. Wat hierbij in feite gebeurt, is dat de metadata die bij de informatie in tabel 1 hoort eenduidig in kaart wordt gebracht. Zo kan alle harde en zachte semantiek die voortvloeit uit de verwoording van alle door de domeindeskundigen aangedragen voorbeelden in de FCO-IM Casetool worden vastgelegd.

Een aldus opgesteld conceptueel informatiemodel wordt in FCO-IM een *informatiegrammatica* (IG) genoemd en kan door de Casetool worden weergegeven in een of meer *informatiegrammaticadiagrammen* (IGD). Figuur 1 toont het IGD behorend bij de IG van de informatie in tabel 1. Omdat bij de analyse netjes gebruik is gemaakt van elementaire zinnen, heet dit een elementair IGD. Zo'n diagram is alleen voor de analist interessant, niet voor de domeindeskundige.

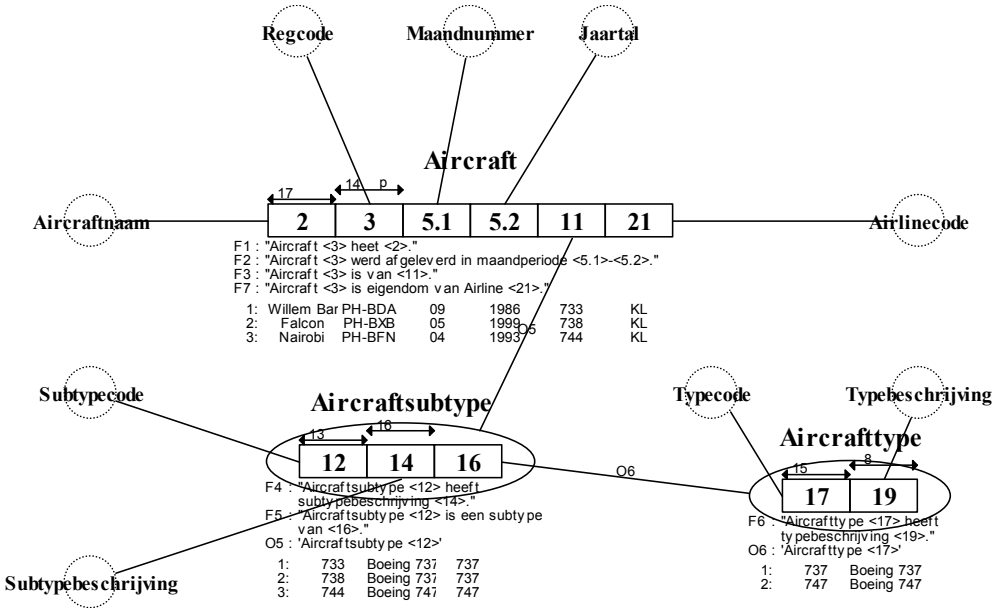


Figuur 1: Elementair IGD van de informatie in tabel 1

Uit de IG bij dit IGD kan met harde wiskundige algoritmen een relationeel databaseschema worden afgeleid. Deze algoritmen heten *Groeperen*, *Reduceren* en *Lexicaliseren*. Omdat een conceptueel model als in figuur 1 zorgvuldig is ontworpen om de gemodelleerde informatie redundantievrij vast te leggen, geldt dat ook voor het relationele databaseschema dat door deze algoritmen wordt gecreëerd. Dat is wiskundig hard bewezen. Het schema is in elk geval in derde normaalvorm (3NF), en als de analist goed zijn best heeft gedaan zelfs in vierde (voor een resultaat in vijfde normaalvorm moet hij nog meer zijn best doen, zie daarvoor [2]).

Bij het groeperen worden zoveel mogelijk feittypen samengevoegd bij één en hetzelfde objecttype, zonder dat daarbij redundantie optreedt. Hierdoor wordt het aantal resulterende objecttypen zo klein mogelijk. Toch kunnen er bij dit proces objecttypen blijven bestaan die best gemist kunnen worden, omdat hun populaties door de opgelegde integriteitregels altijd ook in andere objecttypen voorkomen. Bij het reduceren worden deze

objecttypen alsnog verwijderd, zodat het aantal objecttypen echt zo klein mogelijk wordt. Figuur 2 toont het resultaat van groeperen en reduceren (GR) van het elementaire IGD in figuur 1. Hierin zijn de objecttypen Airline, Maandperiode, Maand en Jaar, die na het groeperen nog wel bestonden, door het reduceren verdwenen. De domeindeskundige had desgevraagd gezegd, dat er geen objecten van deze soort hoefden te worden opgeslagen anders dan die welke al bij Aircraft werden vastgelegd.



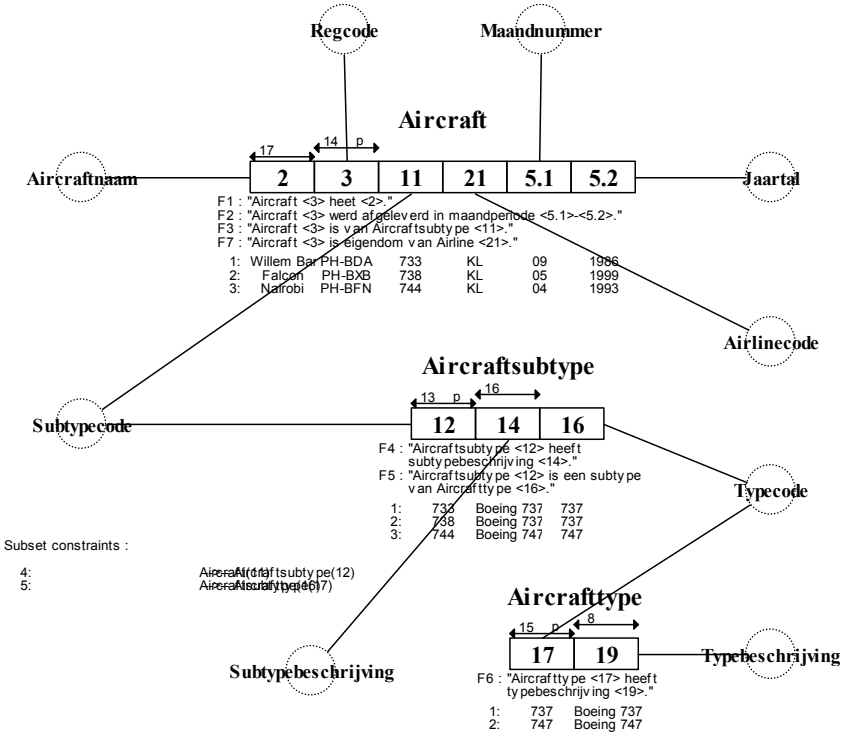
Figuur 2: Het GR-IGD behorend bij het El-IGD in figuur 1

Door het groeperen en reduceren is het informatiemodel op conceptueel niveau omgezet in een model op logisch niveau. Het GR-IGD in figuur 2 is niets anders dan een ER-model in vermomming. De overgebleven objecttypen corresponderen met entiteittypen en de rollen daarin zijn attributen, als ze door een labeltype worden gespeeld. De rollen die door een ander objecttype worden gespeeld corresponderen met een verwijzende (1 op n) relatie tussen de betreffende entiteittypen.

Door het GR-IGD in figuur 2 te lexicaliseren ontstaat een relationeel schema. Hierbij worden de objecttypen zo getransformeerd, dat alle rollen door labeltypen worden gespeeld. Het Relationele Model eist nu eenmaal, dat alle attributen in het model een *lexicaal domein* (een leesbaar datatype) hebben, en in termen van de IGD's in FCO-IM betekent dit dat elke rol gespeeld moet worden door een labeltype. Uiteraard mag ook dit algoritme geen redundantie introduceren. Figuur 3 toont het resultaat van de lexicalisatie (L) van het GR-IGD in figuur 2. Dit 'strokendiagram' is inderdaad een Relationeel Model in vermomming. Elke rol in figuur 3 wordt gespeeld door een labeltype. Omdat nu niet meer aan de objecttypen is te zien dat sommige rollen daarin door andere objecttypen worden gespeeld, is deze informatie in tekst toegevoegd. Subset constraint 4 zegt dat rol 11 in Aircraft verwijst naar rol 12 in Aircraftsubtype, en subset constraint 5 dat rol 16 in Aircraftsubtype verwijst naar rol 17 in Aircrafttype. In relationele taal zijn dit vreemde sleutel verwijzingen. De stroken heten tabellen en de rollen kolommen. Het L-algoritme zet dus het logisch model om in een model op fysiek niveau. Doordat aan elk labeltype al op conceptueel niveau een geschikt datatype is gekoppeld, geldt dat na de GRL-transformaties ook keurig voor elke kolom.

Het zal duidelijk zijn, dat de GRL-transformaties geheel gedreven worden door de harde semantiek: de structuur en beperkingsregels die al op conceptueel niveau in het model zijn vastgelegd. Verder geldt, dat de zachte semantiek (de feittype-expressies die op conceptueel niveau werden vastgelegd) gewoon door deze algoritmen wordt mee getransformeerd. In figuur 1, 2 en 3 zijn steeds dezelfde zintypen te zien (aangegeven door een

F met een cijfer erachter) en ze staan ook steeds bij de juiste objecttypen. In feite vertellen de zintypen in figuur 3 samen met de bijbehorende populaties van de drie tabellen precies het verhaal dat in de database geschreven staat. Hierdoor kan de domeindeskundige achteraf het ontwerp simpelweg valideren door te kijken of de database precies het verhaal vertelt, dat hij er bij de analyse op conceptueel niveau ingestopt wilde hebben.



Figuur 3: Het GRL-IGD behorend bij het El-IGD in figuur 1

Het belangrijkste van dit verhaal is echter, dat de drie GRL-transformaties door de computer kunnen worden uitgevoerd. Een analist die de FCO-IM Casetool gebruikt hoeft daarvoor alleen maar op een knop te drukken. De Casetool kan ook het verhaal dat door het resulterende databaseschema wordt verteld regenereren aan de hand van de gebruikte voorbeeldpopulaties. Figuur 4 toont dit verhaal voor het GRL-IGD in figuur 3. Tenslotte zal duidelijk zijn, dat de Casetool ook gemakkelijk voor elk gewenst platform automatisch een DDL-script kan genereren met alles erop en eraan om het verkregen fysieke model op technisch niveau te implementeren. En dat met het verkregen databaseschema ook het oorspronkelijke voorbeeld in tabel 1 valt terug te genereren.

Expressions generated by FCO-IM Casetool v4.1

Name: AIRCRAFT
Author: Peter W. F. Alons
Description:
Created: 11-09-2000

Expressions for users:

Aircraft:
"Aircraft PH-BDA heet Willem Barentz."
"Aircraft PH-BXB heet Falcon."
"Aircraft PH-BFN heet Nairobi."
"Aircraft PH-BDA werd afgeleverd in maandperiode 09-1986."
"Aircraft PH-BXB werd afgeleverd in maandperiode 05-1999."
"Aircraft PH-BFN werd afgeleverd in maandperiode 04-1993."

"Aircraft PH-BDA is van Aircraftsubtype 733."
"Aircraft PH-BXB is van Aircraftsubtype 738."
"Aircraft PH-BFN is van Aircraftsubtype 744."
"Aircraft PH-BDA is eigendom van Airline KL."
"Aircraft PH-BXB is eigendom van Airline KL."
"Aircraft PH-BFN is eigendom van Airline KL."

Aircraftsubtype:
"Aircraftsubtype 733 heeft subtypebeschrijving Boeing 737-300."
"Aircraftsubtype 738 heeft subtypebeschrijving Boeing 737-800."
"Aircraftsubtype 744 heeft subtypebeschrijving Boeing 747-400."
"Aircraftsubtype 733 is een subtype van Aircrafttype 737."
"Aircraftsubtype 738 is een subtype van Aircrafttype 737."
"Aircraftsubtype 744 is een subtype van Aircrafttype 747."

Aircrafttype:
"Aircrafttype 737 heeft typebeschrijving Boeing 737."
"Aircrafttype 747 heeft typebeschrijving Boeing 747."

Figuur 4: Het verhaal verteld door het databaseschema in figuur 3

Uit het bovenstaande komen de vier basisprincipes waarop FCO-IM is gebaseerd duidelijk naar voren. Deze principes zijn: *100% communicatie georiënteerd, 100% conceptualiteit, 100% valideerbaarheid, en 100% genericiteit*. Deze vier principes zal ik in het vervolg van dit artikel nader beschrijven. Ze zijn essentieel voor het doel dat we bij metadata management willen nastreven. Het zal immers duidelijk zijn, dat bij het transformeren van een informatiemodel naar de verschillende modelleerniveaus niet kan worden verhinderd dat de bijbehorende metadata daarbij redundant worden opgeslagen. Wat we echter willen, is dat daarbij zoveel mogelijk sprake is van *beheerste* redundantie: redundantie die niet leidt tot update-anomalieën. Zoiets treedt ook op bij relationele implementaties van 3NF-modellen. Instanties van sleutels waarnaar verwezen wordt kunnen meer dan eens voorkomen in de database. Deze vorm van redundantie wordt echter volledig beheerst via de opgelegde beperkingsregels, zodat het RDBMS daarbij geen update-anomalieën zal toestaan. Bij het vastleggen van metadata wordt deze beheerste redundantie aangeduid met *'single-point-of-definition'*, en dat is waar we in het vervolg van dit artikel verder op zullen ingaan.

Dit artikel is gepubliceerd in **Database Magazine, nr. 8, jaargang 11, december 2000**.

Referenties

1. **Reengineering the Corporation**, Michael Hammer, James Champy, Harper Business, mei 1994
2. **Volledig Communicatiegeoriënteerde Informatiemodellering**, G. Bakema, J.P. Zwart, H. van der Lek, Kluwer BedrijfsInformatie, 1996

Over de Auteur

Peter Alons, senior consultant bij Origin/Business Management Solutions, is al ruim tien jaar als informatie-analist, project leider en consultant betrokken geweest bij een groot aantal Decision Support en Data Warehouse projecten bij diverse bedrijven. De laatste tijd is hij actief betrokken bij Data Warehouse projecten bij Akzo Nobel, KLM, NS, RABO en ABN-AMRO. Hij vervult daarbij rollen als adviseur, datamodelleur, methode-coach en quality-assurance auditor. Hij geeft presentaties, workshops en colleges over FCO-IM, Metadata beheer en Data Warehousing bij klanten en de Hogeschool van Arnhem en Nijmegen.

Contact

Peter Alons: peter.alons@atosorigin.com
Origin: http://www.atosorigin.com/

Single Point of Definition Metadata (2)

Een betrouwbare weg naar Metadata Management

Peter Alons (Dr. P.W.F. Alons)
Origin/Business Management Solutions (BMS)

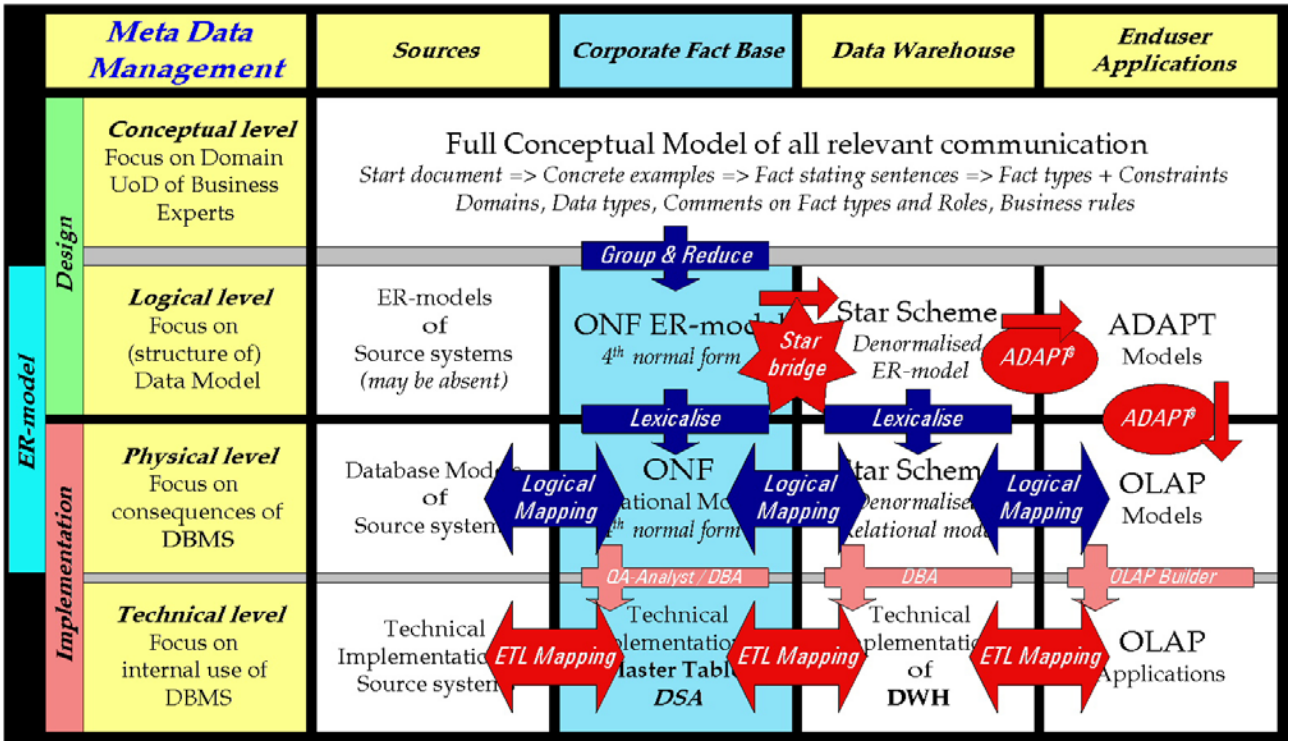
In het vorige artikel heb ik de problemen besproken die het opzetten van adequaat metadata management ernstig bedreigen. Deze lagen enerzijds op het gebied van systeemontwikkeling en anderzijds op het gebied van metadata management zelf. We hebben bij het ontwikkelen van informatiemodellen vier niveaus van modellering onderkend: het *conceptuele*, het *logische*, het *fysieke*, en het *technische* niveau. Daarbij werd benadrukt dat effectief en betaalbaar metadata management vereist, dat deze niveaus goed en zorgvuldig gescheiden worden gehouden. Daarbij bleek het probleem voornamelijk te zitten in het conceptuele niveau, omdat daarvoor in het algemeen slechts zwakke ondersteuning wordt geboden. Als voorbeeld van een methode die dit niveau wel goed ondersteunt werd aan de hand van een concreet voorbeeld FCO-IM behandeld. Daarbij kwamen vier basisprincipes naar voren waarop deze methode is gebaseerd: *100% communicatie georiënteerd, 100% conceptualiteit, 100% valideerbaarheid, en 100% genericiteit*. Deze vier principes werden essentieel genoemd voor het doel dat we bij metadata management nastreven: *beheerste* redundantie bij de opslag van metadata in de verschillende modelleringsniveaus, d.w.z. redundantie waarbij het optreden van update-anomalieën zoveel mogelijk wordt vermeden. Dit principe wordt aangeduid met *'single-point-of-definition'* en wordt in dit artikel nader uitgewerkt. Alvorens daartoe over te gaan geef ik, zoals beloofd, eerst een nadere beschrijving van de vier principes.

1. *100% communicatie georiënteerd:* modelleer *niet* de werkelijkheid zelf, maar de *communicatie* over de werkelijkheid. De werkelijkheid is doorgaans veel te gecompliceerd om vast te leggen en is ook niet waar het om gaat. Daarom wordt in FCO-IM gevraagd representatieve voorbeelden te maken van de informatie die door de domeindeskundigen belangrijk wordt geacht, en deze te verwoorden in zinnen die de feiten in deze voorbeelden tot uitdrukking brengen. Overigens werd het gebruik van concrete informatiedragers in het verre verleden al door Codd aanbevolen.
2. *100% conceptualiteit:* modelleer alle conceptuele aspecten van de communicatie die het informatiesysteem moet ondersteunen, en niets meer of minder dan dat. Hiermee wordt met name bedoeld, dat zaken die op logisch, fysiek of technisch niveau thuis horen op conceptueel niveau geen rol mogen spelen. Dit is internationaal zelfs de standaard maar wordt doorgaans al niet meer geëerbiedigd, als ER-modellering ook voor het conceptuele niveau wordt gebruikt. Vaak worden de inhoudelijke vragen dan afgewisseld met vragen als: "Moeten we hier nu een attribuut of een relatie van maken?". Zo'n vraag kan voor een domeindeskundige natuurlijk nooit 'to-the-point' zijn en hoort dan ook niet op conceptueel niveau thuis.
3. *100% valideerbaarheid:* de domeindeskundigen moeten kunnen vaststellen dat hun communicatie correct is gemodelleerd, zonder dat zij daarvoor het gemaakte model zelf hoeven te inspecteren. In FCO-IM is dit als volgt gegarandeerd. Voor de domeindeskundigen is alleen hun communicatie over de werkelijkheid van belang. Via de verwoordingen van de voorbeelden daarvan krijgt ook de analist inzicht in hoe de domeindeskundigen hun werkelijkheid zien. Omdat de FCO-IM Casetool de gebruikte zinnen na elke transformatie kan teruggenereren, kunnen deze zowel tijdens als na afloop van het modelleren aan de domeindeskundigen worden voorgelegd. Zolang deze bevestigen dat hun relevante informatiedragers juist zijn verwoord, is ook het gecreëerde model correct en anders niet.
4. *100% genericiteit:* alle informatie over het informatiemodel moet worden opgeslagen in één enkele open repository, die generiek is (d.w.z. dezelfde structuur heeft) voor het conceptuele, logische en fysieke niveau. Dit is bij FCO-IM een direct gevolg van het feit, dat de drie typen van diagrammen in figuur 1, 2 en 3 van het voorgaand artikel in

dezelfde diagramtechniek kunnen worden weergegeven. De repository van de FCO-IM Casetool is ontworpen door een volledige verzameling relevante voorbeelden van alle drie de diagramtypen te verwoorden. Fysici zouden de Casetool hierdoor 'zelfconsistent' noemen: als je de verwoording van de drie diagramtypen in de Casetool invoert, dan genereert hij zijn eigen repository als relationeel databaseschema. Bij de uitvoering van de GRL-algoritmen vinden de transformaties in de repository zelf plaats op basis van stapsgewijze wijzigingen in de populatie van deze repository. Bovendien kunnen alle repository updates tijdens deze transformaties rechtstreeks zichtbaar worden gemaakt.

Een Metadata Framework

De vier modelleringsniveaus en de vier basisprincipes die we hierboven hebben genoemd zijn samen te smeden tot een fraai raamwerk voor metadata management: het Metadata Framework. Figuur 1 toont dit raamwerk. Als rijen daarin zien we de vier modelleringsnivaus conceptueel, logisch, fysiek, en technisch, samen met hun indeling in lagen: de ontwerp-, de implementatie-, en de ER-laag. Als kolommen hebben we diverse 'zuilen' van toepassing gekozen. Deze zijn hier toegespitst op Data Warehousing, maar kunnen desgewenst geheel overeenkomstig aan andere soorten van projecten worden aangepast.



Figuur 1: Een Metadata Framework met een aantal belangrijke metadata managementprocessen.

Gemeenschappelijk voor alle toepassingen waarbij sprake is van integratie van informatie over verschillende bedrijfsprocessen heen, is het optreden van de zuil 'Corporate feitenbank'. Hierin moet idealiter sprake zijn van een bedrijfsbreed gegevensmodel dat vrij is van dubbelzinnigheden in de communicatie tussen de diverse bedrijfsprocessen. Nu zijn er in het nabije verleden bij heel wat bedrijven pogingen gedaan om een dergelijk model op te zetten en te onderhouden op het logische niveau, bijvoorbeeld in de vorm van een Corporate ER-model. Deze pogingen zijn echter doorgaans na verloop van tijd doodgebloed. De belangrijkste oorzaak daarvan is, dat bij een dergelijke aanpak op logisch niveau niet alleen de inhoud van het model moet worden onderhouden, maar ook de structuur ervan. Maar als het model inhoudelijk wijzigt, moet als gevolg daarvan vaak ook de structuur van het model ingrijpend worden aangepast. Dit blijkt op den duur voor handmatig werk – dat nu eenmaal hoort te voldoen aan de eisen van volledigheid, correctheid en consistentie (zie ook [3]) - een onoverkomelijk bezwaar te zijn. Dat neemt niet weg, dat ook Ralph Kimball in zijn boeken over Data Warehousing het belang van een dergelijk ER-model en de implementatievormen daarvan

in de Data Warehouse 'Back Room' onderstreept, ook al is ER-modellering overduidelijk niet zijn favoriet.

In de zuil van de bronsystemen zijn natuurlijk ook gegevensmodellen te vinden, in elk geval in de implementatielaag. De bijbehorende modellen op logisch niveau willen wel eens ontbreken, vooral bij echte 'legacy' systemen. In de Data Warehouse zuil willen we graag adequate modellen maken. Zoals uitvoerig is behandeld in de artikelen van Harm van der Lek ([1]), hebben deze in de traditie van Kimball de vorm van een sterschema. Zo'n sterschema valt nog steeds in ER-vorm weer te geven, ook al is er in die modellen sprake van denormalisatie (een ER-tool hoeft daar in al zijn onschuld geen weet van te hebben!). Bij de eindgebruiker-applicaties hebben we ook modellen nodig, en als meest prominente voorbeelden hiervan zijn in het raamwerk de OLAP-modellen aangegeven. Deze kunnen op logisch niveau - met name voor de communicatie van deze modellen naar de eindgebruikers toe - fraai worden weergegeven via een ADAPT-model. ADAPT staat voor "Application Design for Analytical Processing Technologies" en is een ontwerptechniek voor OLAP-modellen, bedacht door Dan Bulos ([2]) en uitvoerig behandeld in de artikelen van Peter Kurstjens ([3]-[5]). Bij Origin/BMS is ADAPT evenals FCO-IM tot standaard verheven.

De clou van het raamwerk in figuur 1 ligt natuurlijk in het conceptuele niveau. Op dat niveau komen alle zuilen feitelijk (een bijzonder treffend woord in deze context!) bij elkaar. Hier spelen specifieke structuurwensen of fysieke en technische implementatiebeslissingen immers nog geen rol. Alles draait hier om welke feiten relevant zijn voor de te ontwikkelen en te onderhouden eindgebruikersapplicaties. En welke feiten relevant zijn en over de verschillende bedrijfsprocessen heen eenduidig geïntegreerd moeten worden, wordt daarbij geheel gedicteerd door de vragen die de eindgebruikers aan hun applicaties willen kunnen stellen (en dus niet - zoals soms wordt gedacht - door alles wat maar in de bronsystemen aanwezig is). Kortom wat hier gevraagd wordt is precies datgene, dat we met FCO-IM heel goed kunnen doen. Voor elke gewenste applicatie stellen we met de betrokken eindgebruikers (de domeindeskundigen) een startdocument op dat de gebruikerswensen weergeeft. We zorgen ervoor dat daarbij concrete voorbeelden op tafel komen, en dat deze verwoord worden in feitpostulerende zinnen. We classificeren en kwalificeren deze zinnen tot elementaire feittypen en leggen de daarbij behorende integriteisregels ('business rules') vast. We zorgen ervoor, dat voor feittypen die in verschillende eindgebruikersapplicaties een rol spelen eenduidige verwoordingen en namen worden gekozen, zodat de voor een Data Warehouse noodzakelijke integratie van informatie kan worden bereikt. We voorzien de labeltypen (domeinen) van geschikte datatypen, en voegen waar maar even wenselijk is commentaren toe bij de feittypen en de rollen die daarin voorkomen. Tenslotte laten we de domeindeskundigen het hele conceptuele model valideren. Als dat alles gebeurd is, staan we aan de grensrivier tussen de conceptuele wereld van de eindgebruikers en de actuele wereld van de informatiesystemen.

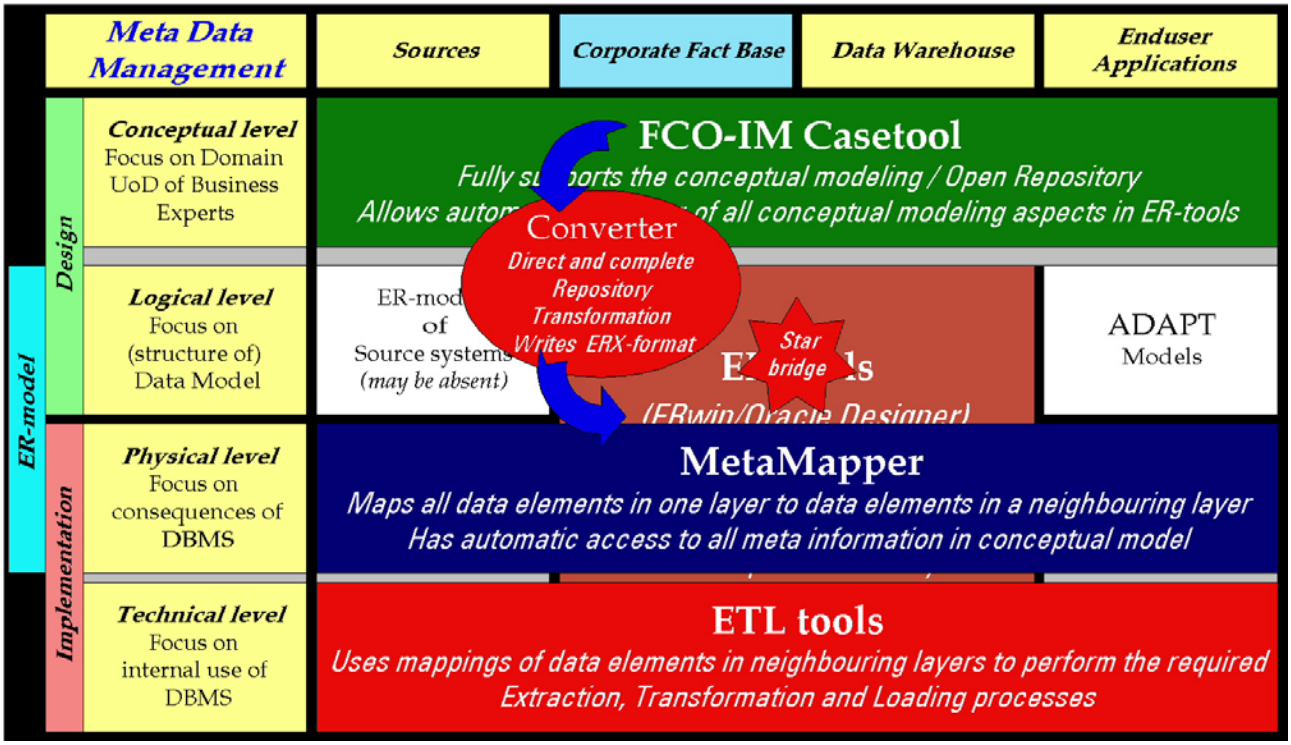
Deze grensrivier is als een extra dikke lijn weergegeven in figuur 1. Waar steken we deze 'Rubicon' over? Waar anders dan in de zuil Corporate feitenbank? Dankzij het GR-algoritme kunnen we immers van het informatiemodel dat we op conceptueel niveau hebben vastgelegd 'met een druk op de knop' een redundantievrij ER-model maken, en evenzo met het L-algoritme een redundantievrij relationeel databaseschema met een zo klein mogelijk aantal tabellen. En dat alles met behoud van alle vastgelegde conceptuele metainformatie. Zo gewenst kan een DBA daar vervolgens een technische implementatie van maken in de 'Data Staging Area' van het Data Warehouse. Hiermee hebben we de eerste stap van ons streven naar beheerste redundantie bij de metadata gedaan. Bij het omvormen van het conceptuele model naar logische en fysieke modellen wordt alle beschikbare en gewenste metadata gekopieerd en dus doorgaans redundant opgeslagen, maar dit wordt niet meer gedaan door mensen. De computer doet dit door de repository van de conceptuele casetool te lezen en harde transformatie-algoritmen te gebruiken. Een niet onbelangrijk neveneffect van dit alles is, dat we de resulterende ER-modellen niet meer nodig hebben voor validatie door de domeindeskundigen, iets waar ze overigens altijd al hoogst ongeschikt voor zijn gebleken.

Het aardige is nu, dat we elke keer als we de applicaties van de eindgebruikers willen uitbreiden hetzelfde kunnen doen. Zodra we de daarvoor zinvolle wijzigingen in de bestaande

feittypen en eventuele nieuwe feittypen hebben ingevoerd, kunnen we de grensrivier opnieuw over door geheel nieuwe modellen te laten genereren, weer met behoud van alle conceptuele informatie. Het is mooi, als we de computer daarbij kunnen laten vaststellen welke *delta's* daarbij zijn opgetreden ten opzichte van de vorige keer. We kunnen daarmee namelijk een tweede stap zetten op weg naar ons doel. Als we om welke redenen dan ook metadata elders redundant hebben opgeslagen, kunnen we de computer deze informatie laten updaten zonder het directe gevaar van update-anomalieën. Een dergelijk proces heet synchronisatie van de metadata, en de tools die dit doen synchronisatie-tools. Vaak moet men dit soort tools in de praktijk nog gedeeltelijk of geheel zelf bouwen, maar als men de onderliggende principes goed begrijpt valt dat werk best mee.

Naast de bovenbeschreven beheerste bewegingen in de zuil Corporate Feitenbank kunnen we ons nu even elegant en beheerst binnen een modelleerniveau bewegen van zuil naar zuil. Zo kunnen we op logisch niveau uit ons ER-model een sterschema laten genereren voor het Data Warehouse met behulp van het 'sterdruk'-algoritme van Harm van der Lek (zie [6]). Uiteraard dient daarbij de nodige geautomatiseerde ondersteuning te worden geboden. Door het sterschema te lexicaliseren kan daar weer automatisch een implementeerbaar databaseschema bij worden gemaakt, dat door een DBA kan worden geïmplementeerd. Waar gewenst kan bij dit alles gebruik worden gemaakt van incrementele synchronisatie. Evenzo kunnen we aan de hand van het sterschema ADAPT-modellen maken, die zijn om te zetten in OLAP-modellen, waarvan een OLAP-Builder vervolgens OLAP-applicaties kan maken. Deze laatste lijn wordt op dit moment nog niet optimaal computerondersteund, maar daar wordt bij Origin/BMS wel aan gewerkt.

Op het fysieke niveau kunnen we met behulp van synchronisatie-tools mappings maken, die vertellen hoe de velden in de eindgebruikersapplicaties samenhangen met de velden in het sterschema en met de velden in het Corporate Feitenbank model. Uiteraard kunnen we daarbij volledig gebruik maken van alle informatie die op conceptueel niveau is vastgelegd. Tenslotte zouden we ook nog kunnen aangeven aan welke bronvelden de betreffende gegevens onttrokken worden. Al dit soort informatie zouden we computergestuurd via het intranet aan de gebruikers beschikbaar kunnen stellen. Het is juist op dit vlak waar synchronisatietools hun grote nut bewijzen. Tenslotte kunnen we ons evenzo op het technische niveau laten helpen met het genereren van mappings voor ETL-tools, waarbij we ons en passant nog geheel aan de in de literatuur aanbevolen architecturen kunnen houden.



Figuur 2: Het Metadata Framework uit figuur 1 met ondersteunende tools voor metadata managementprocessen.

In figuur 2 is het raamwerk van figuur 1 nog eens getoond, dit keer met een beeld van alle computer-tools die we daarbij kunnen inzetten. Prominent daarin zijn - naast de positie van de FCO-IM Casetool - de 'Converter', de 'Star bridge' en de 'MetaMapper', die wij bij Origin/BMS hebben laten ontwikkelen. Ter wille van ruimtebesparing zwaai ik hier anoniem lof toe aan allen die daaraan hebben bijgedragen. De Converter leest de repository van de FCO-IM Casetool na toepassing van de GR-algoritmen en schrijft alle informatie daaruit in een file met ERX-formaat, die gelezen kan worden door ER-tools als Power Designer en Erwin. Hierdoor kan een gewenste ER-omgeving geheel automatisch worden gevuld met een model dat op conceptueel niveau ontworpen is, en wel met een snelheid, rijkdom en precisie, die realistisch gesproken handmatig niet haalbaar zijn (en dit komt in aanmerking voor 'understatement of the year'). De Star bridge biedt een analist de mogelijkheid om maximaal computerondersteund uit een 3NF ER-model een optimaal sterschema te genereren. De MetaMapper is in zijn huidige vorm een tool in MS Access die kan vaststellen welke incrementele wijzigingen er in het Corporate ER-model en het daaruit afgeleide sterschema zijn opgetreden bij elke wijziging van het model. De tool neemt daarbij allerlei meta-informatie die op conceptueel niveau is vastgelegd over, en maakt het mogelijk (incrementeel) vast te leggen hoe de velden in de verschillende modellen met elkaar samenhangen. Een Data Administrator kan aan de hand van dit alles allerlei nuttige metadata rapporten genereren, terwijl een eindgebruiker via een browser informatie kan krijgen over hoe de velden in het sterschema samenhangen met de velden in het Corporate ER-model en welke definities er bij die velden zijn vastgelegd.

De 'Wrap up'

In dit en het voorgaande artikel heb ik als stelling geponeerd dat beheersbaar, en daarmee verantwoord en betaalbaar metadata management vereist, dat de vier niveaus van modellering - conceptueel, logisch, fysiek en technisch - goed en zorgvuldig gescheiden worden gehouden. Ik heb getracht daarbij te beschrijven wat ik daar precies mee bedoel. Het ultieme doel daarvan is om 'single-point-of-definition' metadata management mogelijk te maken, waarmee redundante opslag van de metadata optimaal beheersbaar kan worden gehouden. In de praktijk zijn er nu twee situaties te onderscheiden.

Meestal zien we in grote databaseprojecten modelleringstrajecten waarin geen gebruik wordt gemaakt van technieken als FCO-IM, oftewel trajecten met een zwakke ondersteuning op conceptueel niveau. Dit leidt doorgaans tot de volgende bronnen van serieuze problemen:

1. Zwakke ontwerpen, onder andere als gevolg van weinig betrouwbare validaties door de domeindeskundigen.
2. Handmatige vastlegging van het logisch model in een ER-omgeving. Mensen kunnen daarbij fouten maken, zelfs als ze correcte input proberen te geven.
3. Niet beheerste redundante opslag van alle metadata. De vraag is bijvoorbeeld, hoe en waar alle oorspronkelijke begripsmatige uitgangspunten zijn vastgelegd en de hoe consistentie daarvan met de andere modelleringsniveaus wordt bewaakt.

Daartegenover staan aanpakken waarbij bijvoorbeeld gebruik wordt gemaakt van FCO-IM en dus van volledige ondersteuning op conceptueel niveau. Dit heeft in elk geval deze directe gevolgen:

1. De eerstgenoemde bron van problemen is sterk verminderd. Er kunnen nu goede ontwerpen worden gemaakt, mede door het gebruik van een volledig betrouwbare validatietechniek door de domeindeskundigen;
2. De tweede bron van fouten kan geheel vermeden worden, waardoor de ER-modellen op logisch niveau volledig consistent zijn met de modellen op het conceptuele niveau;
3. De noodzaak om onbeheerste redundante opslag van metadata toe te laten kan geminimaliseerd worden. De metadata kunnen zoveel mogelijk 'single-point-of-definition' worden gehouden.

Het gaat te ver om te claimen dat 100 % Single Point of Definition Metadata daarmee een haalbare kaart wordt. Maar door nog eens te kijken naar alle problemen rond metadata management die in het begin van het eerste artikel zijn opgesomd kunt U zelf vaststellen in welke mate een volledige ondersteuning op conceptueel niveau, zoals beschreven, en het voorgestelde Metadata Framework een oplossing bieden voor deze problemen. In ieder geval kan ik

zeggen, dat deze aanpak in een van de laatste Data Warehouse projecten waaraan ik heb meegewerkt een werkelijk effectief en efficiënt metadata management mogelijk heeft gemaakt. En daarmee kunnen we terug langs de weg, waarlangs ik dit artikel begon. Betaalbaar, betrouwbaar en efficiënt metadata management is op zijn minst een zegen voor het op langere termijn betaalbaar houden van het onderhoud aan een Data Warehouse en levert daarmee een effectieve bijdrage tot het slagen van de totale Data Warehouse inspanningen.

Dit artikel is gepubliceerd in **Database Magazine, nr. 1, jaargang 12, februari 2001**.

Referenties

1. **Sterren en dimensies**, Dr. H. van der Lek, F. Habers, M. Schmitz, Array Publications, 1999
2. **OLAP Database Design: A New Dimension**, Dan Bulos, Database Programming & Design, Vol. 9 no. 6, juni 1996
3. **OLAP-modellering moet over andere boeg**, Peter Kurstjens, DB/M jaargang 10 no. 3, mei 1999
4. **OLAP-modellering moet over andere boeg (2)**, Peter Kurstjens, DB/M jaargang 10 no. 4, juni 1999
5. **OLAP-modellering moet over andere boeg (3)**, Peter Kurstjens, DB/M jaargang 10 no. 5, september 1999
6. **Op jacht naar de sterren**, Harm van der Lek, DB/M jaargang 11 no. 2, april 2000

Over de Auteur

Peter Alons, senior consultant bij Origin/Business Management Solutions, is al ruim tien jaar als informatie-analist, project leider en consultant betrokken geweest bij een groot aantal Decision Support en Data Warehouse projecten bij diverse bedrijven. De laatste tijd is hij actief betrokken bij Data Warehouse projecten bij Akzo Nobel, KLM, NS, RABO en ABN-AMRO. Hij vervult daarbij rollen als adviseur, datamodelleur, methode-coach en quality-assurance auditor. Hij geeft presentaties, workshops en colleges over FCO-IM, Metadata beheer en Data Warehousing bij klanten en de Hogeschool van Arnhem en Nijmegen.

Contact

Peter Alons: peter.alons@atosorigin.com
Origin: http://www.atosorigin.com/

