

## OLAP Modelling met behulp van ADAPT™

Naar aanleiding van het artikel zoals verschenen in Database Magazine (09/1999) met als titel  
"OLAP modellering moet over een andere boeg"



Peter Kurstjens

## Table of Contents

OLAP Modeling .....	3
OLAP objecten.....	4
ERD & DFD modellen .....	5
Ster schema's .....	5
Introductie ADAPT .....	6
Dimensies .....	7
Tijddimensie .....	8
Feit dimensie.....	8
Hyperkubussen en contexten .....	9
Modelleringregels .....	10
Modelleren van Dimensies .....	10
Modelleren van Calculaties .....	11
Modelleren van Members en Scopes .....	11
Conclusie.....	12
Over de auteur .....	12
Referenties .....	12
Contact .....	12

## OLAP modellering

De laatste jaren heeft de Online Analytical Processing (OLAP) technologie in hoog tempo zijn weg weten te vinden naar de desktop van een groeiende groep gebruikers. Een tiental jaar geleden nog was OLAP voorbehouden aan data analisten en controllers, die met gespecialiseerde software geavanceerde analyses maakten van vooral financiële en markt gerelateerde gegevens.

In de financiële sector moet u hierbij denken aan toepassingen als consolidatie van financiële gegevens van een multinational, routinematige berekening van een groot aantal financiële controle parameters (IFO, RONA, etc.), investeringsanalyses, etc. etc. Financiële applicaties worden doorgaans gekenmerkt door (relatief) beperkte gegevens verzamelingen, en een groot aantal -soms complexe- berekeningen.

In de marketing wereld werd en wordt OLAP gebruikt voor de analyse van (soms zeer) grote verzamelingen verkoopgegevens: om de markt te kunnen onderverdelen (bijv. naar afzetkanalen en doelgroepen), en om inzicht te krijgen in het koopgedrag van die doelgroepen. Dit type applicatie werkt dan wel vaak op zeer omvangrijke gegevensverzamelingen, maar de berekeningen die plaats moeten vinden zijn vaak veel simpeler: aggregatie, en het berekenen van ratio's (bijv. marktaandeel).

De gebruikte software gereedschappen waren vaak erg krachtig, maar vrij moeilijk toegankelijk, en vereisten een forse leerinspanning. Naast kennis van de software was ook wiskundige kennis onontbeerlijk, omdat de aanwezige gereedschapkast in deze tools zonder veel (intelligente) verpakking ter beschikking werd gesteld aan de eindgebruiker. U moet denken aan SAS, SystemW, Oracle Express, HoloS. We zagen dan ook dat alleen business analisten met de software werkten, terwijl het management genoeg nam met vrij statische rapportages. Het compromis was het Executive Information System, waarmee in feite de statische rapportages werden ingepakt in een applicatie die eenvoudige navigatie (waaronder slice and dice faciliteiten) bood.

De laatste jaren zien we echter dat dankzij leveranciers als Cognos, Business Objects en anderen on-line data analyse voor grotere groepen toegankelijk wordt. Onder druk van deze toetreders hebben ook de 'traditionele' leveranciers de instapdrempel verlaagd met een verbeterde User Interface of met 'lichte' versies van hun pakketten. Niet alleen managers worden uitgenodigd zelf met de gegevens te 'spelen', maar alle zgn. kenniswerkers in een onderneming worden geacht met behulp van dit type OLAP applicatie het inzicht in het reilen en zeilen van de onderneming te kunnen vergroten. Dat kan direct bijdragen aan de verbetering van bedrijfsprocessen, maar vergroot ook de betrokkenheid van de medewerkers bij de onderneming.

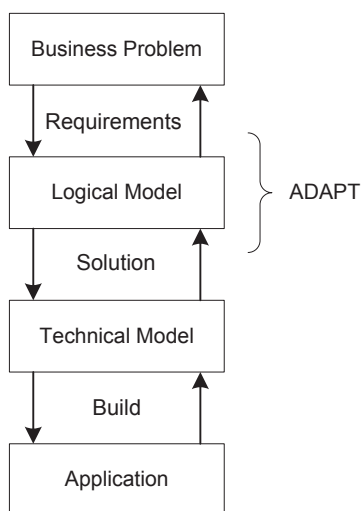
Het belang van dat laatste aspect weegt, zeker in tijden van krapte op de arbeidsmarkt, steeds zwaarder.

De zgn. desktop OLAP tools hebben van de geavanceerdere familieleden de multidimensionale concepten overgenomen -met slice en dice en drill up/down/across functionaliteit-. Maar waar deze software in veel mindere mate in voorziet is de calculatie functionaliteit nodig voor consolidatie, berekening van ratio's en dergelijke. Ook kunnen ze vaak slechter overweg met uitzonderingen in de gegevens structuur. De functionaliteit van deze software is dus veel meer gericht op het presenteren en rapporteren van gegevens in een multi-dimensionale structuur, dan het analyseren daarvan. We kunnen wel vaststellen, dat onder druk van de markt en de complexe praktijk ook de functionaliteit van deze populaire tools voortdurend wordt uitgebreid.

En daarmee is ook de behoefte toegenomen naar methodes en technieken om OLAP applicaties te ontwerpen. Niet langer is het een kleine groep van gespecialiseerde medewerkers of consultants die verantwoordelijk is voor het ontwerp, de realisatie en het onderhoud van OLAP systemen: een steeds grotere groep informatie analisten en automatiseerders komt in aanraking met dit type applicatie. De ervaring heeft geleerd dat het ook met eenvoudige hulpmiddelen als desktop OLAP tools niet moeilijk is in korte tijd een volkomen chaotische en onbeheersbare situatie te creëren. Een goede ontwerpmethodede is noodzakelijk, om de juiste gegevens en functionaliteit te bieden in de OLAP applicaties, en deze onderling consistent te houden.

Doorgaans bieden leveranciers van OLAP producten middels handleidingen of cursussen zelf wel een 'methode' om een OLAP applicatie te ontwerpen. Maar deze methode is dan wel sterk toegespitst op het product in kwestie. Dat betekent dat de methode bepaalde beperkingen van het product zal trachten te vermijden of te compenseren. En dat leidt dan vervolgens in het beste geval tot een minder logisch ontwerp; in het slechtste geval worden bepaalde aspecten van de bedrijfssituatie compleet genegeerd! Dit gevaar is bij de desktop OLAP tools meer prominent aanwezig, omdat deze tools ten opzichte van de geavanceerdere familieleden een duidelijk beperktere functionaliteit bieden.

Als we er wat langer bij stil staan is een ontwerpmethodede die gekoppeld is aan een software tool sowieso problematisch: een ontwerp zou moeten helpen bij het selecteren van een software tool. Het ontwerp dient in de eerste plaats zo goed mogelijk de bedrijfsproblematiek te weerspiegelen, als ook de wensen van de beoogde gebruikers van de OLAP applicatie. Vervolgens kan het ontwerp gebruikt worden bij de selectie van een (set van) software tool(s) die optimaal aansluit bij de bedrijfsproblematiek.



Figuur 1: Positionering ADAPT

We hebben dus behoefte aan een ontwerpmethode die los staat van de beschikbare tools. Waarom dan niet een reeds bestaande en bewezen methode hiervoor gebruikt? In dit artikel zal ik laten zien, dat bestaande veelgebruikte ontwerpmethoden, zoals ERD en DFD niet voldoen bij het ontwerpen van OLAP applicaties. Tot die conclusie is ook Dan Bulos, van Symmetry Corporation, gekomen. Hij heeft gedurende de afgelopen jaren gewerkt aan een nieuwe methodiek, ADAPT genaamd, waarvan de fundamenten vanaf hoofdstuk 'Introductie ADAPT' aan de orde zullen komen.

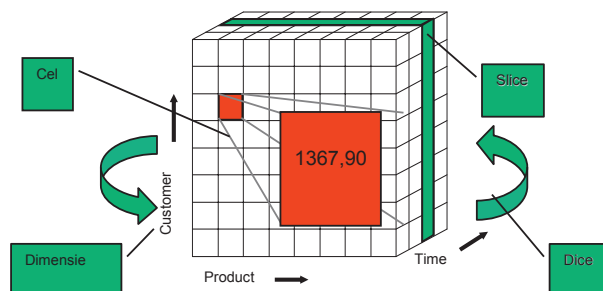
Voor we naar de modelleringstechnieken gaan kijken is het goed vast te stellen wat we willen bereiken met onze modelleringsinspanning. Een goed model voldoet aan de vier C's:

- **Completeness:** zijn alle belangrijke aspecten van de applicatie gemodelleerd? Het is daarbij van ondergeschikt belang hoe gedetailleerd e.e.a. is gemodelleerd.
- **Correctness:** zijn alle belangrijke aspecten ook correct gemodelleerd. D.w.z. kan men met de modellen in de hand verifiëren dat het model ook voldoet aan de eisen?
- **Consistency:** is het model intern consistent? Consistentie is belangrijk voor een model, omdat dit de begrijpelijkheid van het systeem enorm verhoogt en de realisatie sterk kan vereenvoudigen.
- **Communication:** is de essentie van de OLAP applicatie in het model zodanig gepresenteerd, dat die overdraagbaar is naar alle partijen die bij het ontwerp, de realisatie of onderhoud zijn betrokken (en dat is dus inclusief de eindgebruiker!)?

## OLAP objecten

Wat zijn nu de objecten die we willen kunnen modelleren in een OLAP applicatie?

Om die vraag te kunnen beantwoorden hoeven we maar te kijken naar een aantal OLAP tools. We vinden in al deze tools de multi-dimensionale kubus, of hyperkubus als basis object voor de presentatie van gegevens terug. De hyperkubus is dan ook een onmisbaar object in een OLAP model. Omdat wij mensen er moeite mee hebben om ons kubussen met meer dan drie dimensies voor te stellen is het gebruikelijk om een driedimensionale kubus te tekenen.



Figuur 2: OLAP concepten

Met deze presentatie metafoor hebben we direct een tweede onmisbaar object geïdentificeerd: de dimensie. De dimensie kan nu eenvoudig worden gedefinieerd als een unieke, samenhangende verzameling elementen die zinvol langs een as in de hyperkubus kunnen worden afgebeeld. Uniek, zodat elementen maar in een enkele dimensie kunnen vóórkomen, en samenhangend, zodat het voor de eindgebruiker ook nog begrijpelijk blijft. Het element van een dimensie is natuurlijk zélf ook een object van de applicatie, vooral van belang om uitzonderingen te kunnen modelleren. En uitzonderingen hebben vaak betrekking op méér dan een enkel element, dus ook een set van elementen kan als separaat object worden onderkend.

Een belangrijk object in elke OLAP applicatie is de hiërarchie. De hiërarchie brengt orde aan in de elementen van een dimensie. De hiërarchie levert daarnaast een navigatie pad voor drill up en drill down operaties, en (dus) ook de aggregatie logica.

Verder kunnen we nog rekenmodellen onderkennen, waarmee allerlei afgeleide gegevens kunnen worden berekend. Aan deze rekenmodellen wordt vaak ten onrechte weinig aandacht geschonken in de eerste fasen van een OLAP project. Dat is riskant, omdat, zoals al eerder is aangegeven, veel met name desktop OLAP tools erg veel moeite hebben met complexere berekeningen. Het komt dus geregeld voor dat pas later in een project komt vast te staan of het gebruikte tool in staat is om bepaalde berekeningen uit te voeren. En dan hoeft u echt niet direct te denken aan complexe statistische analyses: de meeste desktop OLAP tools zijn al niet in staat om in een berekening (bijv. aggregatie) met uitzonderingen om te gaan. Dat leidt dan tot problemen als u bijv. de resultaten van een zojuist verworven minderheidsdeelname buiten het totaal wenst te houden.

## ERD & DFD modellen

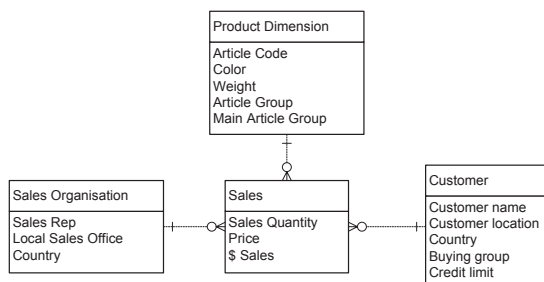
De meest gebruikte technieken voor het modelleren van applicaties zijn ongetwijfeld Entity Relationship Diagramming en Data Flow Diagramming. ERD diagrammen zijn bijzonder geschikt voor het modelleren van datastructuren die in een relationele database zullen worden opgeslagen. De methode leidt via een aantal normalisatieslagen tot een ontwerp dat voor transactie georiënteerde systemen erg geschikt is, omdat het geen redundantie meer kent. DFD diagrammen beschrijven de processen waarmee de data bewerkt wordt.

De tekortkomingen van ERD technieken voor het modelleren van multidimensionale systemen zijn al door vele auteurs aan de orde gesteld (onlangs nog door Harm van der Lek, in dit blad). Kortweg komt het er op neer, dat OLAP systemen geheel andere karakteristieken hebben dan transactie systemen. Mutaties in de systemen vinden vrijwel altijd volledig plaats met behulp van batch processen, die vaak maar eenmaal per maand lopen (hoewel de update cyclus van OLAP databases de neiging heeft steeds korter te worden zal die toch zelden vaker dan eenmaal per dag worden uitgevoerd). Er is dan ook meestal maar één proces dat de updates verzorgt. Dat betekent dat er in de ontwerpen nauwelijks rekening gehouden hoeft te worden met multi-user updates in de database. De belangrijkste uitzondering op deze 'regel' wordt gevormd door de budgettering- en planning-systemen, waar doorgaans wél meerdere gebruikers gelijktijdig gegevens inleggen.

Aan de andere kant zullen gebruikers van OLAP systemen vaak grote hoeveelheden data moeten benaderen om antwoord op bepaalde vragen te kunnen krijgen. Dat geldt natuurlijk voor data mining toepassingen, maar ook voor een simpel Executive Information System, dat omzetgegevens op het topniveau van een grote organisatie moet presenteren. Met name deze laatste categorie vragen is zeer voorspelbaar, en kan alleen efficiënt worden opgelost indien redundant geaggregeerde gegevens worden opgeslagen. Ook in bij de opslag van de dimensie structuren is vaak redundante opslag noodzakelijk, vooral om het aantal joins in een query te minimaliseren.

## Sterschema's

Ster schema's zijn waarschijnlijk de bekendste en succesvolste ERD modellen voor de data opslag van OLAP applicaties, en dan met name voor de modellering van het Data Warehouse. Een ster-schema is weergegeven in Figuur 3.



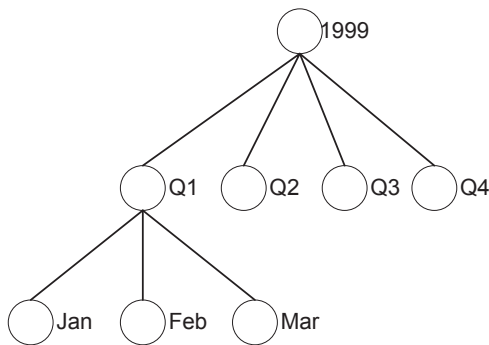
Figuur 3: ERD Ster Schema

Binnen in de ster zijn de zgn. Feittabellen gemodelleerd, eromheen zijn de dimensies gemodelleerd. De feittabellen bevatten de te analyseren gegevens –dus de gegevens in de hyperkubus-, en zijn middels foreign key relaties gekoppeld aan de dimensietabellen. Als we een dergelijk model bestuderen, zien we dit model wel enige informatie verschaft over de OLAP applicatie, maar dat vele van de eerder genoemde OLAP objecten nog erg in mist blijven hangen:

- De hiërarchieën zijn niet te herkennen in de dimensietabellen. Dat is erg vervelend, omdat zoveel karakteristieken van een OLAP applicatie afhangen van het aantal en de structuur van de hiërarchieën. Zo hangt de totale omvang van een volledig doorgerekende OLAP database sterk af van het aantal hiërarchieën, het aantal levels in iedere hiërarchie en het aantal elementen op ieder level. Denk echter ook eens aan de complexiteit die wordt veroorzaakt door het werken met verschillende kalenders (Gregoriaans, Fiscaal, Productie,...). Deze kalenders worden vaak als alternatieve hiërarchieën binnen de tijddimensie gemodelleerd.
- (Verzamelingen van) Elementen van een dimensie zijn evenmin gemodelleerd. Dit maakt het absoluut onmogelijk om uitzonderlijke situaties die afhangen van de inhoud van de dimensietabellen te modelleren (bijv. kennen vele organisaties een de afwijkende organisatie structuur in de Verenigde Staten t.o.v. Europa, om met staten i.p.v. landen te kunnen werken).
- Van de feittabellen kan niet –eenvoudig- worden aangegeven op welk aggregatieniveau de gegevens nu precies worden ingelegd. Het is immers eerder regel dan uitzondering, dat niet alle gegevens voor een OLAP database op hetzelfde nivo worden aangeleverd. Zo worden budgetten meestal op een hoger niveau ingelegd dan de verkopen. En als dat nog is op te lossen met behulp van verschillende feittabellen, dan wordt het toch al een stuk lastiger, indien alleen Brazilië de budgetgegevens op een hoger aggregatie niveau mag aanleveren.
- Rekenmodellen worden in het geheel niet gerepresenteerd. De correcte berekening van bijv. Performance Indicatoren is echter toch cruciaal voor elke OLAP applicatie. Het onvermogen om rekenmodellen te kunnen weergegeven uit zich bijv. duidelijk in het tijddomein, waar constructies als Moving Annual Total (MAT), Year To Date (YTD) totalen, of verschillen tussen gelijke maanden van verschillende jaren vrijwel altijd in OLAP applicaties vóórkomen.
- De ERD methode geeft geen enkele richtlijn over het correct opbouwen van een ster-schema. Hoe bepalen we wat de dimensies moeten zijn, de hiërarchieën?

De completering van de ERD schema's met DFD schema's brengt geen soelaas. Met behulp van DFD's kunnen uitstekend de batchprocessen worden gemodelleerd waarmee de OLAP database wordt geüpdate, maar ze zijn duidelijk niet in staat zelfs simpele constructies als Moving Annual Total te modelleren.

Deze bezwaren moeten zwaar wegen, omdat ze het vrijwel onmogelijk maken dat een eindgebruiker het model op juistheid en volledigheid kan beoordelen. Tegelijkertijd leiden deze tekortkomingen ertoe dat erg veel informatie moet worden vastgelegd en overgedragen in tekst en zelfbedachte schetsjes, zoals bijv. in:



Figuur 4: Tijd hiërarchie

Naast deze functionele tekortkomingen van ERD en DFD technieken is er nog een overweging die het gebruik voor het modelleren van OLAP applicaties in de weg staat. De lezers die aandachtig de artikelen van Harm van der Lek hebben gevolgd, of de boeken van Ralph Kimball tot zich genomen hebben, zullen hebben ontdekt dat er vele slimme optimalisaties mogelijk zijn op een eenvoudig ster schema, die de performance dramatisch kunnen verbeteren, en/of de opslagbehoeften van de applicatie belangrijk kunnen beperken. Het valt buiten de scope van dit artikel om hier diep op in te gaan, maar als we als voorbeeld naar de introductie van zogenaamde mini dimensies kijken, dan zien we dat deze dimensies uitsluitend om technische redenen worden gecreëerd. Deze dimensies zijn dus voor eindgebruikers irrelevant, en leiden tot verwarring.

Kijken we –samenvattend– naar de vier criteria die ik graag wil hanteren voor een goed model, dan móet de combinatie van ERD en DFD schema's wel haast ernstig tekort schieten:

- Completeness: bepaalde belangrijke objecten (hiërarchieën, elementen, verzamelingen, calculatiemodellen) worden niet of maar deels gemodelleerd.
- Correctness: de correctheid van een set ERD en DFD schema's is niet te beoordelen, wegens een gebrek aan informatie.
- Consistency: het handhaven van logische consistentie wordt een probleem als bijv. bij de introductie van minidimensies.
- Communication: het gebrek aan informatie over specifieke OLAP objecten beperkt het communicatieve nut van de schema's enorm.

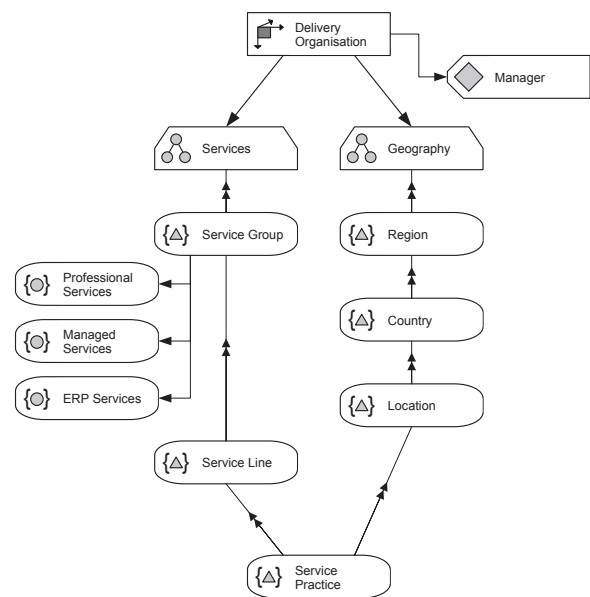
De conclusie moet zijn, dat ERD en DFD diagrammen nog redelijk voldoen bij het (technische) ontwerp van het data warehouse, maar dat zij niet voldoen voor het modelleren van data marts en de OLAP applicaties die er toegang toe hebben.

In het vervolg van dit artikel zal ik een tool-onafhankelijke ontwerpmethodiek – ADAPT – introduceren, die door Dan Bulos van Symmetry Corporation speciaal voor OLAP applicaties is ontwikkeld.

ADAPT staat voor: Application Design for Analytical Processing Technologies.

ADAPT is speciaal ontwikkeld om de belangrijkste aspecten van een OLAP applicatie op eenduidige wijze logisch te modelleren. ADAPT biedt dan ook een tekenset en werkwijze die nauw aansluit bij de problematiek van OLAP applicaties.

Figuur 5 geeft een indruk van de expressieve kracht van een ADAPT ontwerp. Het is een vereenvoudigd schema van de organisatie structuur van een dienstverlenend bedrijf. Het leveren van diensten is belegd bij gespecialiseerde Service Practices, die enerzijds in een Services hiërarchie (Service Line – Service Group), anderzijds in een Geografische hiërarchie (Location – Country – Region) zijn opgehangen. Elke nivo in de organisatie kent een manager, wat eenvoudig wordt aangegeven door een attribuut te definiëren voor de gehele dimensie (gekoppeld aan Delivery Organisation). Verder duidt het schema aan dat er drie Service Groups bestaan: Professional, Managed en ERP Services.



Figuur 5: ADAPT schema Organisationsdimensie

## Introductie ADAPT

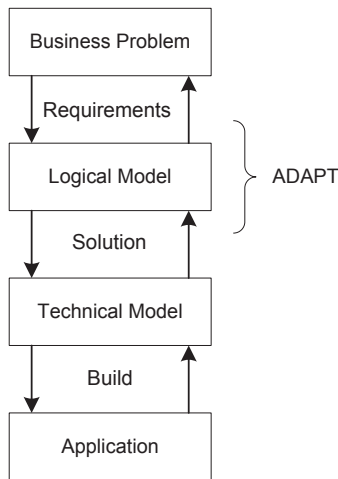
Bovenstaand heb ik een aantal ontwikkelingen beschreven die de behoefte aan een ontwikkelmethode voor OLAP applicaties sterk heeft doen toenemen in de afgelopen jaren.

Het blijkt dat voor het ontwerpen van OLAP applicaties momenteel in de praktijk een mix wordt gebezigd van ERD en DFD methoden, aangevuld met een leverancier- of ontwerperspecifieke aanpak en presentatiewijze.

Deze schieten alle serieus tekort, als het gaat om het bereiken van een ontwerp dat voldoet aan de eisen van completeness, correctness, consistency en communication. In het vervolg van dit artikel zullen we een tool-onafhankelijke ontwerpmethodiek – ADAPT – introduceren, die door Dan Bulos van Symmetry Corporation speciaal voor OLAP applicaties is ontwikkeld.

ADAPT staat voor: Application Design for Analytical Processing Technologies.

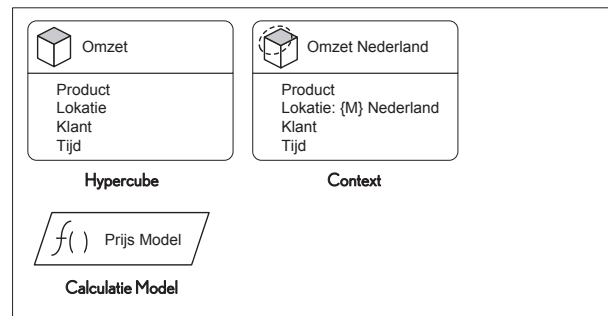
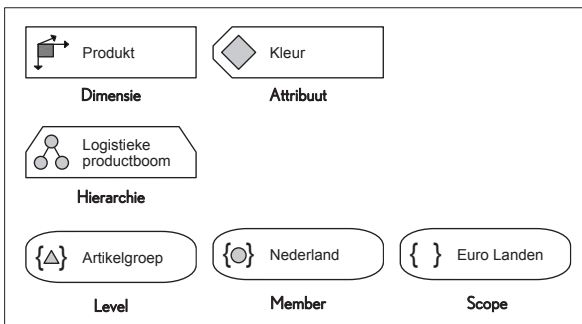
ADAPT is speciaal ontwikkeld om de belangrijkste aspecten van een OLAP applicatie op eenduidige wijze logisch te modelleren. ADAPT biedt een tekenset en werkwijze die nauw aansluit bij de problematiek van OLAP applicaties.



Figuur 6: Positionering ADAPT

Het zal dan ook geen verbazing wekken, dat alle OLAP objecten direct te vertalen zijn naar ADAPT symbolen.

Laat ik deze symbolen eerst introduceren, om daarna aan de hand van een aantal voorbeelden duidelijk te maken hoe die met behulp van deze symbolen bepaalde ontwerpbeslissingen duidelijk kunnen worden uitgedrukt.

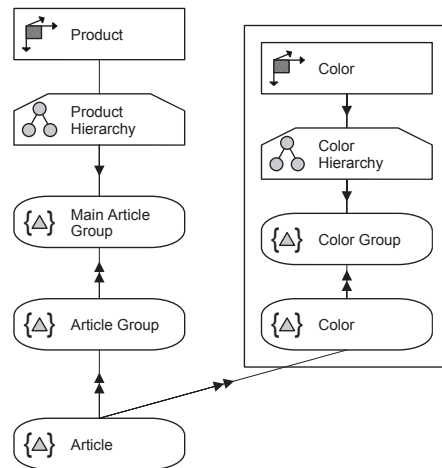


Figuur 7: ADAPT Objecten

## Dimensies

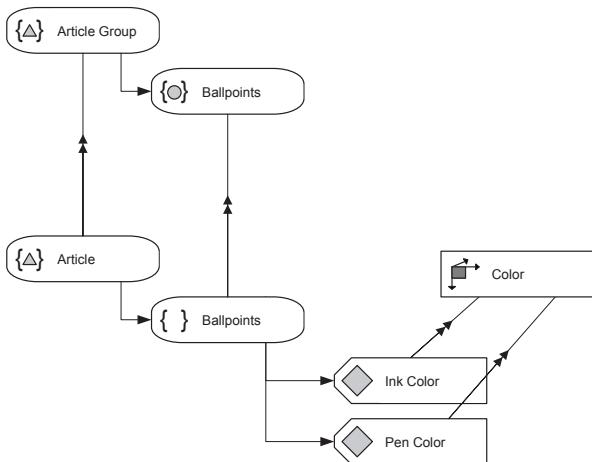
In Figuur 8 is een nog vrij eenvoudige Product dimensie afgebeeld. De figuur geeft aan dat binnen de Product dimensie een enkele hiërarchie is gedefinieerd, met drie hiërarchische niveaus. De dubbele pijl naar boven geeft het aggregatiepad aan. De artikelen hebben een kleur, en door een aparte kleur dimensie te definiëren geven we aan dat er bij de eindgebruikers behoefte bestaat gegevens te analyseren door kleur en producten in een matrix tegen elkaar uit te zetten. Dat is dan vooral zinvol voor het Article Group en Main Article Group niveaus.

Omdat kleur een eigenschap is van artikelen, kan de kleur dimensie worden geclassificeerd als een property dimensie. De Product dimensie is niet op een dergelijke manier afhankelijk van een andere dimensie, en wordt geclassificeerd als een core dimensie.



Figuur 8: Product Dimensie

De werkelijkheid is echter zelden zo eenvoudig. In staat een wat realistischer detail afgebeeld. Een leverancier van kantooraccessoires kent wellicht een Article Group "Ballpoints". De artikelen die tot deze Article Group behoren kennen twee kleurattributen, namelijk de kleur van de ballpoint en de kleur van de inkt.



**Figuur 9: Modelleren van uitzonderingen**

Daarmee vormt "Ballpoints" een uitzondering t.o.v. de andere Article Groups, en wordt "Ballpoints" expliciet getekend als Member van het level Article Group.

Op het Article niveau kunnen we de ballpoints modelleren, door een Scope (verzameling van Members) te definiëren, genaamd "Ballpoints". De hiërarchische relatie tussen deze artikelen en de Article Group "Ballpoints" wordt middels de dubbele opwaartse pijl aangegeven. Alléén de artikelen in deze Scope hebben de attributen "Ink Color" en "Pen Color". En de relatie naar de Color Dimensie geeft aan dat gebruikers de producten via beide attributen op kleur willen kunnen analyseren.

Het valt op, dat we in dit schema de inkt en pen kleur expliciet als attribuut hebben gemodelleerd. In Figuur 8 is het "artikelkleur" attribuut niet expliciet gemodelleerd, maar wordt impliciet aangeduid door de dubbele pijl naar het "Color" level uit de kleur dimensie. In Figuur 9 is de kleur dimensie niet uitgewerkt. Dat dient geïnterpreteerd te worden als een relatie naar het laagste niveau van de kleur dimensie.

In dit voorbeeld is er maar een enkele kleurdimensie gedefinieerd. Daarin zijn dus zowel de penkleuren als de inktkleuren opgenomen. De analyse van product versus kleur zijn nu duidelijk alléén zinvol voor de Ballpoints. We zien dat we met ADAPT uitstekend in staat zijn om deze complexe situatie op een compacte wijze eenduidig te modelleren.

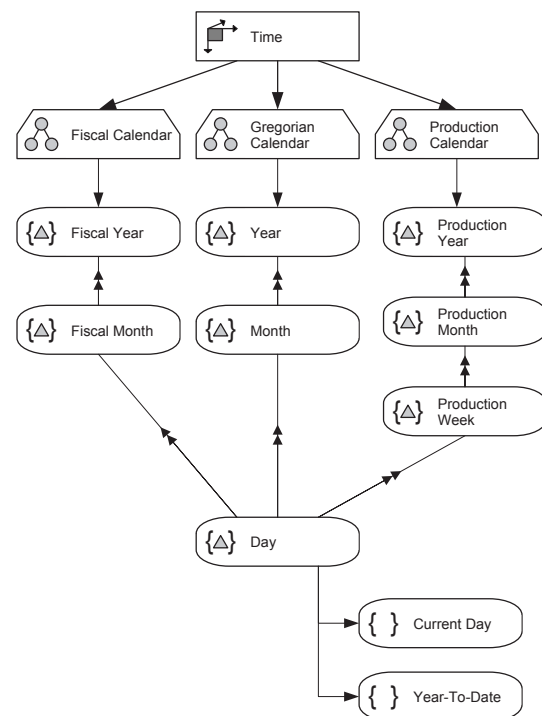
### Tijd dimensie

De tijd dimensie is een belangrijke component in vrijwel elke OLAP applicatie. Deze dimensie is ook in een aantal aspecten uitzonderlijk:

1. De standaard kalenderstructuur is volledig vóórgedefinieerd.
2. Er worden vaak alternatieve structuren gedefinieerd, bijv. om met ploegendiensten te kunnen omgaan, of een afwijkende fiscale kalender te accommoderen.

3. De tijddimensie is sequentieel. Dat wil zeggen dat er een logische volgorde bestaat tussen de members van die dimensie. In een typische OLAP applicatie is deze volgorde van groot belang: er wordt bijv. gewerkt met Moving-Annual-Total totaal laatste twaalf maanden), Year-to-Date (jaartotaal tot en met huidige datum) en allerlei vergelijkingen in de tijd (huidige maand versus zelfde maand vorig jaar, maandelijkse groeipercentages, etc.)

In Figuur 10 is een voorbeeld van een tijddimensie gegeven. In dit voorbeeld zijn drie onafhankelijke hiërarchieën gemodelleerd. Wat nieuw is in dit model zijn de dynamische scopes "Current Day" en "Year-to-Date". Deze scopes zijn dynamisch, omdat de samenstelling van deze scopes voortdurend -in de tijd- verandert. Op deze manier kan worden aangegeven welke tijdconstructies van belang zijn in de OLAP applicatie.



**Figuur 10: Tijd Dimensie**

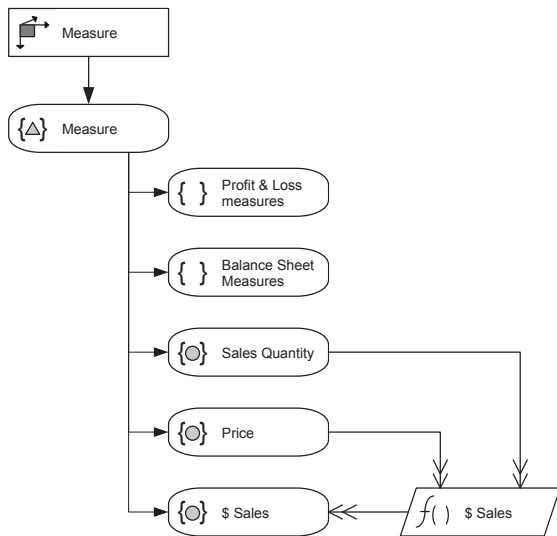
### Feitdimensie

Een tweede dimensie die in iedere OLAP applicatie terugkeert is de feit ("measure") dimensie. Ook deze feit dimensie heeft een aantal duidelijke karakteristieken:

- Doorgaans is er geen hiërarchie gedefinieerd (al kan dat voor m.n. financiële applicaties nog wel van toepassing zijn).
- Er is vaak een (groot) aantal calculaties tussen de members van deze dimensie gedefinieerd.
- Deze calculaties kunnen erg complex zijn.

Een eenvoudige measure dimensie is afgebeeld in Figuur 11. Nieuw in deze figuur is het calculatiemodel. Dit model geeft aan dat de omzet uit verkochte eenheden en prijs.

Wat opvalt is dat dit model uitsluitend de afhankelijkheden aan- geeft tussen de members. Dat is een bewuste keuze vóór inzicht en overzicht, wat ten koste gaat van het detail. Om een ontwerp te kunnen doorgronden is het niet noodzakelijk exact te weten hoe een rekenformule is opgebouwd. Wél van belang is de het inzicht in de onderlinge afhankelijkheden van de measures. Dat geeft ons inzicht in de noodzaak tot het verzamelen en vastleggen van be- paalde basisgegevens en de volgorde van berekeningen. Nadere detaillering wordt dus verwezen naar aanvullende, tekstuele docu- mentatie van het model.



Figuur 11: Feit Dimensie

## Hyperkubussen en contexten

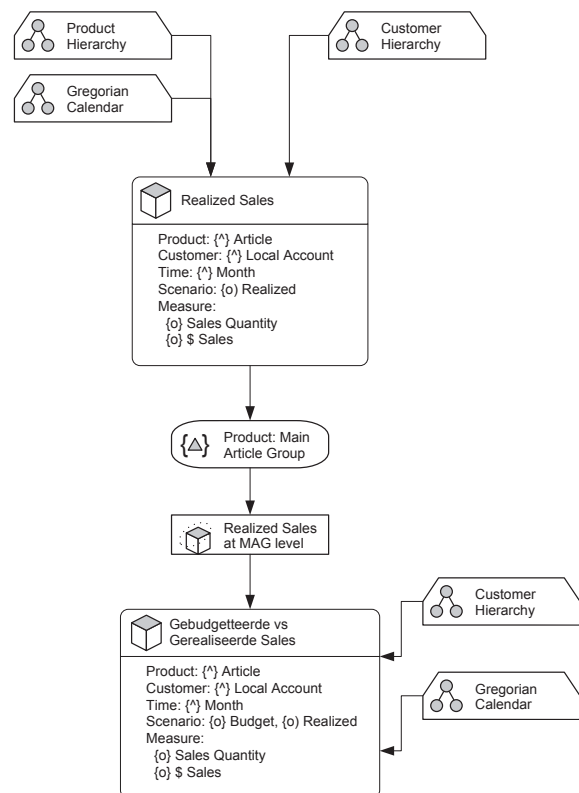
In de hyperkubus worden de gegevens opgeslagen van de OLAP applicatie. De wijze van presenteren in een ADAPT schema –zie Figuur 12, de 'Realized Sales hyperkubus'- is erg krachtig, omdat niet alleen wordt aangegeven door welke dimensies de kubus wordt gedefinieerd ('vorm'), maar ook op welk aggregatieniveau de gegevens worden vastgelegd ('omvang'). Het model in het voorbeeld geeft ook nog aan voor welke volgens welke hiërarchie-ën de hyperkubus geaggregeerde gegevens bevat.

Er zijn een groot aantal verschillende aggregatieregels mogelijk. De meest vóórkommende functie is uiteraard de sommatie. Maar een OLAP ontwerper dient zich ervan bewust te zijn dat voor bepaalde measures andere aggregatieregels gehanteerd moeten worden. Zo zal een voorraadniveau (een 'balans' measure) in de tijd niet geaggregeerd kunnen worden door te sommeren, maar door de eerste of laatste waarde te nemen van de periode (dit is typerend voor sequentiële dimensies). Een ander bekend voor- beeld is het aggregeren van prijzen, waar met (gewogen) gemid- delden gewerkt moet worden.

Ook geavanceerde functies als het alloceren van budgetten die op een hoog niveau zijn ingelegd naar lagere aggregatie niveaus kan van belang zijn. In ADAPT kan de aggregatiefunctie worden ver- meld in de pijl tussen het Hiërarchie- en het Hyperkubus-symbool.

Een doorsnee OLAP applicatie zal meerdere hyperkubussen tel- len, omdat een een hyperkubus uniforme gegevens bevat: geve- gens die voor dezelfde dimensies en op dezelfde aggregatieni- veaus zijn gedefinieerd.

Zo zullen budgetten doorgaans niet op het laagste product- en/of klant-niveau worden bepaald. Dat betekent dat gebudgetteerde omzetgegevens in een andere hyperkubus moeten worden opge- slagen dan de gerealiseerde gegevens. Indien dan toch gebudget- teerde en gerealiseerde omzet met elkaar vergeleken en geanaly- seerd moeten worden, dan zullen de gegevens bij elkaar gebracht moeten worden, op eenzelfde niveau. Ook dit is in Figuur 12 weer- gegeven. Daartoe wordt de Context "Realized Sales at Main Arti- cle Group level" gedefinieerd als een subset van de "Realized Sales" kubus, door de geaggregeerde gegevens op MAG-niveau te selecteren. Deze context is van dezelfde vorm en omvang als de gebudgetteerde gegevens, en kan dus aan deze hyperkubus wor- den toegevoegd. Aggregatie volgens de Product Hierarchy is voor deze hyperkubus is niet erg zinvol, en is dus weggelaten.



Figuur 12: Sales Hyperkubus

## Modelleringsregels

Nu we een beetje gevoel hebben ontwikkeld voor hoe een ADAPT schema eruit ziet, en wat ermee gemodelleerd kan worden, is het tijd om de aandacht te richten op de moeilijke vragen, zoals: hoe gaan we te werk om een goed ontwerp te maken?

Welke stappen moeten we doorlopen? Waar liggen de valkuilen? Wat willen we nog modelleren in het logische model, wat laten we over voor nadere detaillering in het technisch ontwerp?

De eerste regel moet zijn:

*Start met het business probleem en eindig bij het business probleem.*

Onderzoek naar bronsystemen is noodzakelijk, zoals we verderop in dit artikel zullen laten zien, maar het essentieel voortdurend het doel van de gehele exercitie in het oog te houden, en ook de gesprekspartners hiervan te doordringen. Zondigen tegen deze regel betekent vrijwel altijd dat het ontwerpproces verdrinkt in een vloedgolf van mogelijkheden.

Een goede tweede basisregel is:

*Modelleer precies zoveel als nodig is om de applicatie niet te laten falen. Voeg detail toe in een iteratief ontwerpproces.*

Het ontwerp moet compleet, correct en consistent zijn, maar bovenal de communicatie tussen de betrokkenen ondersteunen. Deze 4 C's laten nog veel vrijheid toe een detailleringniveau te kiezen waarbij de geïnvesteerde energie in modellering nog een duidelijk positief rendement heeft.

Zo zal het in financiële applicaties met honderden measures ('concept codes') vaak niet nodig zijn deze measures individueel te modelleren: veel kan gemodelleerd worden met behulp van goed gekozen scopes, zoals de "{} Profit and Loss concept codes".

De derde regel:

*Begin met dimensies (en hiërarchieën) te modelleren. Negeer in eerste instantie de uitzonderingen.*

Hoewel het voor een onervaren ontwerper vaak een triviale exercitie lijkt, is het in de praktijk vaak bijzonder lastig om een consistente set dimensies te definiëren waar alle gebruikersgroepen achter kunnen staan. Het is erg belangrijk dat de totaalset consistent is, omdat een de OLAP applicatie(s) anders de verwarring in de organisatie eerder doen toenemen dan afnemen. De visie van verschillende gebruikers(groepen) moet dus op elkaar afgestemd worden.

De vierde regel:

*Onderzoek de data bronnen.*

Onderzoek aan de databronnen kan helpen bij het vinden van dimensies en hiërarchieën, maar is ook belangrijk om inzicht te krijgen in de niveau's waarop gegevens beschikbaar zijn, en de

kwaliteit van die gegevens (compleetheid, tijdigheid, consistentie, betrouwbaarheid).

De vijfde regel:

*Modelleer allereerst de generieke rekenregels. Negeer in eerste instantie de uitzonderingen.*

Rekenregels op uitzonderingen kunnen het overzicht doen verliezen. Voeg deze dus pas toe als de generieke rekenregels duidelijk zijn.

Veel members en scopes worden dan pas relevant en dus toegevoegd aan de dimensie structuren.

## Modelleren van Dimensies

De eerste -en niet te onderschatten- stap in het modelleren van een OLAP applicatie betreft de dimensies. Het streven moet zijn een éénduidige set van dimensies te vinden, waarmee alle gebruikerswensen kunnen worden geadresseerd.

Dimensies worden vaak in een aantal iteraties gevonden, waarin beurtelings de gebruikers worden geraadpleegd en de gegevensbronnen worden geanalyseerd.

Het gesprek met de gebruiker is essentieel, omdat uiteindelijk de analysebehoefte van de eindgebruikers bepalend moeten zijn voor het OLAP ontwerp. Het probleem is echter vaak dat eindgebruikers onvoldoende zicht hebben op de datastructuren om samenhangende dimensies te definiëren. In gesprekken met meerdere gebruikers uit de doelgroep kan gemakkelijk een vrij lange lijst met kandidaat-dimensies ontstaan. Vaak zijn deze dimensies echter niet allemaal onafhankelijk van elkaar, maar komen ze op lagere detailniveaus samen in core dimensies, en blijken veel dimensies dus eigenlijk property dimensies te zijn.

Zo kan een marketing manager spreken over dimensies als "leeftijdscategorie", "geografie" en "inkomenscategorie", wat allemaal property dimensies zijn van de core dimensie "customer". Omdat de marketingmanager geen analyses doet tot op het niveau van de individuele customer zal hij van het bestaan van deze core dimensie geen melding doen.

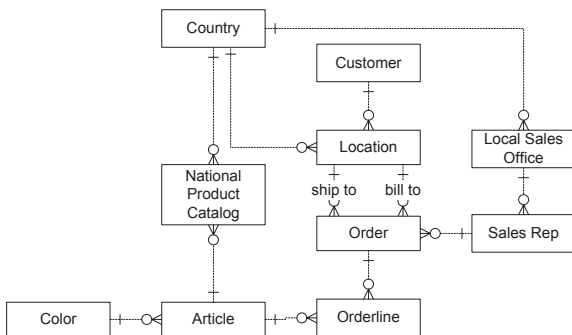
In dit voorbeeld zijn de property dimensies nog vrij gemakkelijk te onderkennen, maar in de praktijk is dat lang niet altijd even eenvoudig. Stel dat in het gegeven voorbeeld ook de eigen organisatie langs geografische lijnen is georganiseerd. Dan is het niet direct duidelijk of de marketingmanager de omzet aan bepaalde gebruikersgroepen wil analyseren volgens de woonplaats van de gebruiker ('land van de klant') of volgens de vestigingsplaats van de verkooporganisatie ('land van de organisatie').

Om zicht te krijgen op de core dimensies, is het erg nuttig om de gegevensbronnen te analyseren. Als we -in een relationele database- de tabellen kunnen opsporen waar de measures worden opgeslagen, dan zijn de foreign key relaties goede kandidaten voor core dimensies.

Als voorbeeld is een vereenvoudigd ERD van een orderregistratiesysteem weergegeven in Figuur 13.

De interessantste gegevens vinden we in de orderline en order tabellen, waar we kunnen achterhalen welke producten tegen welke prijs door welke Sales Rep zijn verkocht aan welke (lokale vestiging van de) klant. Daarmee hebben we de core dimensies Time, Product, Sales Organization en Customer al direct gevonden. We kunnen verder onderkennen, dat Color een kandidaat property dimensie is op de Product dimensie, net als Country op de Customer dimensie. Minder duidelijk is de rol van Country in de Sales Organization dimensie.

We zien een duidelijke parentchild relatie tussen Country en Local Sales Office. Daarmee kan Country ook worden beschouwd als een hoger aggregatieniveau in de Sales Organization dimensie. Hier zal de gebruiker een uitspraak moeten doen.



**Figuur 13: Order Registratiesysteem**

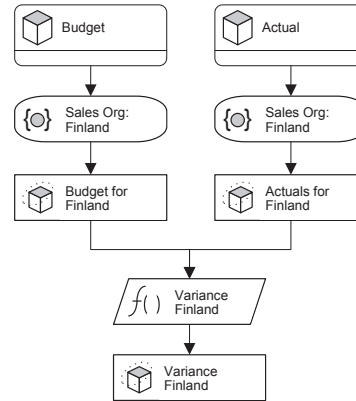
Ook bestudering van de gegevensbron kan snel leiden tot een explosie van potentiële dimensies. In principe kan namelijk vrijwel ieder attribuut van elke tabel leiden tot een (property) dimensie in de OLAP applicatie. Denk bijv. aan de leeftijdscategorie dimensie van de marketingmanager, die wordt opgebouwd met behulp van het geboortedatum attribuut in de customer tabel. Het is dus zaak om de bevindingen af te stemmen met de eindgebruikers, en alleen die dimensies op te voeren die daadwerkelijk voor analyses van belang worden geacht. Een dergelijke overweging geldt ook voor de keuze van hiërarchieën: er zijn vele kandidaten, maar welke hiërarchieën zijn echt relevant?

### Modelleren van Calculaties

Calculaties kunnen op twee manieren worden gemodelleerd: als operaties tussen members van eenzelfde dimensie óf als operaties tussen hyperkubussen.

Een voorbeeld van de eerste categorie is de aggregatie langs hiërarchische lijnen. Een ander voorbeeld is het prijscalculatie model gegeven in Figuur 11. De inter-dimensie modellen worden geïnterpreteerd als generieke rekenmodellen. In principe kunnen deze modellen worden toegepast op alle kubussen waarin de inputwaarden van de berekening voorhanden zijn.

Niet alle berekeningen kunnen op deze manier worden gemodelleerd. Indien de vereiste gegevens niet voorhanden zijn in een enkele hyperkubus, of indien de berekening alléén van toepassing is op een selectie van een hyperkubus (context) zal een intrakubus calculatie uitgevoerd moeten worden, zoals weergegeven in Figuur 14.



**Figuur 14: Intra-kubus calculaties**

### Modelleren van Members en Scopes

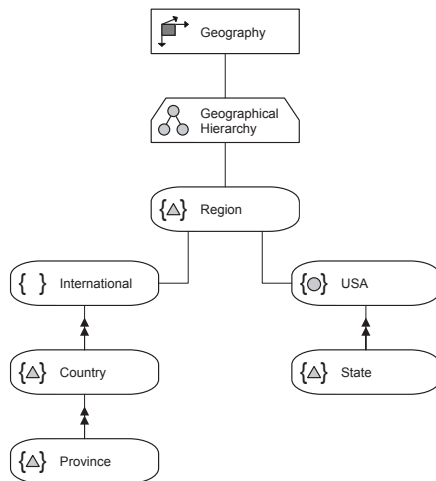
In het algemeen is het slechts nuttig individuele members te modelleren als:

- Het een klein aantal opsombare members betreft (bijv. geografische regio's: Asian Pacific, North and South America, Europe). Het opsommen kan structuren verduidelijken, en de complexiteit en omvang van een OLAP applicatie aangeven.
- Er uitzonderingen zijn verbonden aan de members. Zoals in een eerder voorbeeld de artikelen van de Article Group "Ballpoints" een aantal specifieke en belangrijke attributen kennen.

Scopes vormen de krachtigste gereedschappen in de ADAPT modellering. Welbeschouwd zijn dimensies, hiërarchieën en levels bijzondere verschijningsvormen van scopes: ze impliceren alle een set van members.

*Iedere set kan een subset zijn van een andere set.*

Dat betekent dat een scope ook het startpunt kan zijn van een (sub) hiërarchie, die alléén van toepassing is voor members in een bepaalde scope, zoals in Figuur 15 is weergegeven.



Figuur 15: Mixed scopes

## Conclusie

In dit artikel heb ik een nieuwe methode geïntroduceerd, die het een stuk eenvoudiger maakt om OLAP applicaties te ontwerpen, en de ontwerpen te toetsen op completeness, correctness, consistency, en communication.

De methode is in de praktijk ontwikkeld door Dan Bulos van Symmetry Corporation, een klein gespecialiseerd adviesbureau in de Verenigde Staten.

De methode wordt door de consultants van Nippur en een groot aantal opgeleide consultants al ruim vijf jaar toegepast, en op zijn merites beoordeeld. Reden waarom Nippur door middel van trainingen deze methodologie wil delen met haar collega's en klanten.

Peter Kurstjens, Nippur

## Over de auteur

Peter Kurstjens, managing consultant bij Nippur, is gedurende meer dan tien jaar als architect betrokken bij een groot aantal OLAP en Data Warehouse projecten bij multinationale ondernemingen. Momenteel is hij als adviseur werkzaam voor een aantal Philips organisaties, waaronder Corporate Control, Consumer Electronics en Corporate IT, waar de architectuur en ondersteunende infrastructuur voor de management informatie en rapportage systemen opnieuw worden gedefinieerd. Zijn ruime ervaring met OLAP applicaties, ook in minder voor de hand liggende omgevingen zoals in combinatie met shop floor control systemen, hebben geleid tot een leidende rol in het modelleren van complexe OLAP applicaties.

Voor opmerkingen of vragen kunt U Peter Kurstjens bereiken via onderstaande contactgegevens.

## Referenties

Dan Bulos: A New Dimension, Database Programming & Design, Volume 9, Nr. 6.

## Contact

Nippur: [info@nippur.nl](mailto:info@nippur.nl)

Peter Kurstjens: [peter.kurstjens@nippur.nl](mailto:peter.kurstjens@nippur.nl)

Symmetry Corporation: [www.symcorp.com](http://www.symcorp.com)

Nippur: [www.nippur.nl](http://www.nippur.nl)