Gestructureerde aanpak biedt houvast voor technisch ontwerp

Modelgedreven ontwerp van ETL-functies (2)

Mark Zwijsen

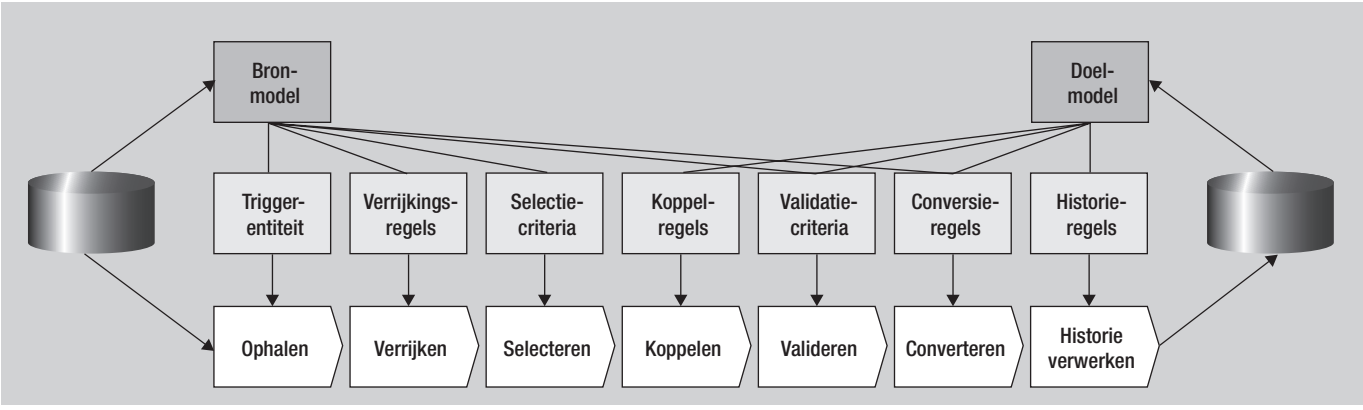
In het artikel 'Modelgedreven ontwerp van ETL-functies' in DB/M 6 van 2004 is een methode beschreven voor het functioneel beschrijven van het ETL-proces. In dit vervolgartikel wordt verder ingegaan op de vertaling van dit functioneel ontwerp naar een technisch ontwerp of technische implementatie.

Een korte samenvatting volgt van het eerste artikel, gepubliceerd in DB/M 6, oktober 2004. Analoog aan het ontwikkelingsproces van de gegevensstructuur van een datawarehouse kan ook in het ontwikkelingsproces van de ETL-functies onderscheid gemaakt worden tussen een logisch ontwerp, dat los staat van de gekozen tool of het platform, en een technisch ontwerp, dat een tool-specifieke implementatie is van het logische ontwerp.
De gepresenteerde aanpak voor het logische ETL-ontwerp hanteert een onderverdeling van de transformatie in een beperkt aantal deelfuncties, die transformatiestappen worden genoemd:
1. Ophalen; 2. Verrijken; 3. Selecteren; 4. Koppelen; 5. Valideren; 6. Converteren. De output van elke stap (zie afbeelding 1) is kort omschreven in afbeelding 2.

Voorbeeld

Bij het verder uitwerken en toelichten van de vertaling naar een technische implementatie en naar testgevallen, wordt gebruik gemaakt van een voorbeeld. Dit voorbeeld betreft het ETL-proces voor telefoongespreksfeiten, met de meetwaarden GespreksDuur en GespreksOmzet. Deze feitentabel is in afbeelding 3 weergegeven als sterschema met de bijbehorende dimensies.
De gespreksfeiten worden gevoed uit de CDR-tabel (CDR = Call Detail Record) van het Customer Care systeem. Van een gesprek is bekend tussen welke twee telefoonnummers het is gevoerd. Een telefoonnummer kan in de loop der tijd toegekend worden aan een andere aansluiting. Een aansluiting is gekoppeld aan een klant. Het relevante deel van het datamodel is weergegeven in afbeelding 4.
De transformatieregels, zoals deze per stap en aan de datamodellen van bron en doel zijn gerelateerd, kunnen overzichtelijk in tabelvorm worden weergegeven. In afbeelding 5 zijn de transformatieregels voor de gespreksfeiten opgenomen.

De koppeling met de dimensies Kalender en Gesprekscategorie zijn in deze uitwerking weggelaten. Indien nodig kan ook aggregatie worden gespecificeerd in de transformatie-regels. Het ligt voor de hand dat er méér dan 1 gesprek kan horen bij een specifieke combinatie van (datum, klant, categorie). Daarvoor kan



Afbeelding 1: Transformatiemodel.

Transformatiestap	Output
OPHALEN	Alle kandidaten met de impliciet aanwezige attributen
VERRIJKEN	Alle kandidaten met de direct *en indirect* gekoppelde attributen
SELECTEREN	Alle *relevante* kandidaten met de direct *en indirect* gekoppelde attributen
KOPPELEN	Alle *relevante* kandidaten met de direct *en indirect* gekoppelde attributen, *plus de verwijzende sleutelattributen van het doelmodel*
VALIDEREN	Alle *relevante en kwalitatief goedgekeurde* kandidaten met de direct *en indirect* gekoppelde attributen, *plus de verwijzende sleutelattributen van het doelmodel*
CONVERTEREN	Alle *relevante en kwalitatief goedgekeurde* kandidaten *met alle benodigde attributen voor het doelmodel*
HISTORIE VERWERKEN	Valt buiten het bestek van dit artikel

Afbeelding 2: Resultaat van de functionele transformatiestappen.

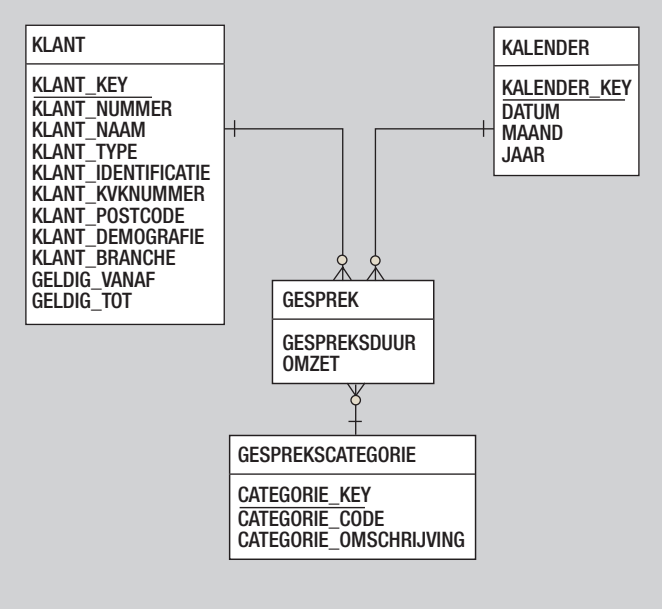
bij de transformatiestappen 12 en 13 (converteren) worden gespecificeerd op welke wijze dient te worden geaggregeerd: 'GESPREKSDUUR = SOM(CDR.DURATION)'.

Vertaling naar technische implementatie

Is het hierboven beschreven ETL-ontwerp nog platform-onafhankelijk, uiteindelijk moet een platform-specifieke vertaling gemaakt worden. Dit komt er vaak op neer dat met behulp van de 'designer'-component van het ETL-product een 'mapping' gebouwd wordt. Bij het bouwen van deze mapping spelen de volgende ingangsgegevens een rol:

1. De functionele regels zoals hierboven beschreven;
2. Beperkingen en mogelijkheden van de ETL-tool;
3. Standaarden en richtlijnen met betrekking tot naamgeving, herbruikbaarheid, structuur, etcetera;
4. En uiteraard de efficiëntie van het uiteindelijke proces (de performance).

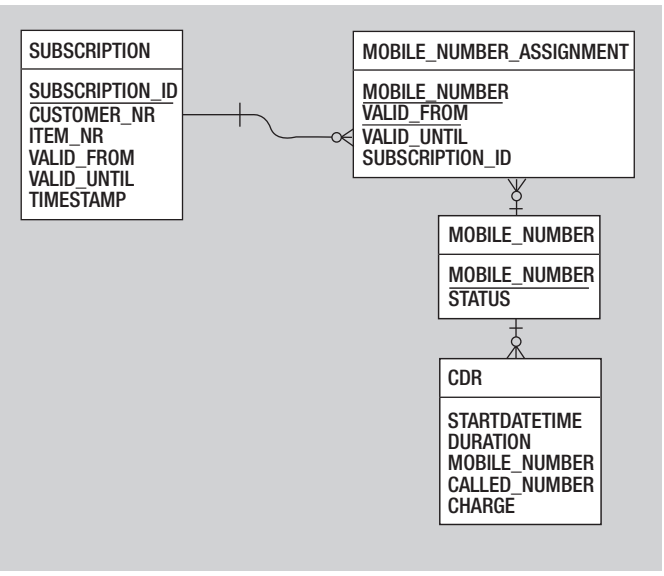
De beperkingen en mogelijkheden beïnvloeden de wijze waarop deze functionele regels worden gerealiseerd. Zo zal bijvoorbeeld een functionele 'koppeling' in de ene tool gerealiseerd kunnen worden met een 'lookup'-transformatie, terwijl dit in de andere tool niet mogelijk is en er gebruik moet worden gemaakt van een 'join'-transformatie. In het geval van het verwerken van een type-2 mutatie in een dimensie, waarbij op basis van 1 aangeleverd record er zowel een update als een insert moeten plaatsvinden, kan er in de ene tool gekozen worden voor een 'pivot'-transformatie, terwijl in de andere tool een 'splitter'-transformatie wordt gebruikt.
Daarnaast zal een streven naar een zo efficiënt mogelijk



Afbeelding 3: Sterschema.

verwerkingsproces ook invloed hebben op de technische inrichting. Hierbij valt te denken aan het zo vroeg mogelijk filteren van gegevens. In afbeelding 6 is een mogelijke uitwerking in een mapping diagram weergegeven.

RDR1 is de 'reader' die de CDR's leest (ophalen#1) en alleen die CDR's selecteert die gisteren zijn ontstaan (selecteren#5).
LKP1 is de 'lookup' die het bijbehorende record in MOBILE_NUMBER zoekt (verrijken#2).
RTR1 is de 'router' die niet geslaagde verrijkingen routeert naar de reject-tabel (validatie#8) en die bij een geslaagde verrijking filtert op de status (selectie#6).
Voor LKP2, RTR2, LKP3 en RTR3 gelden soortgelijke beschrijvingen.



Afbeelding 4: Datamodel brongegevens.

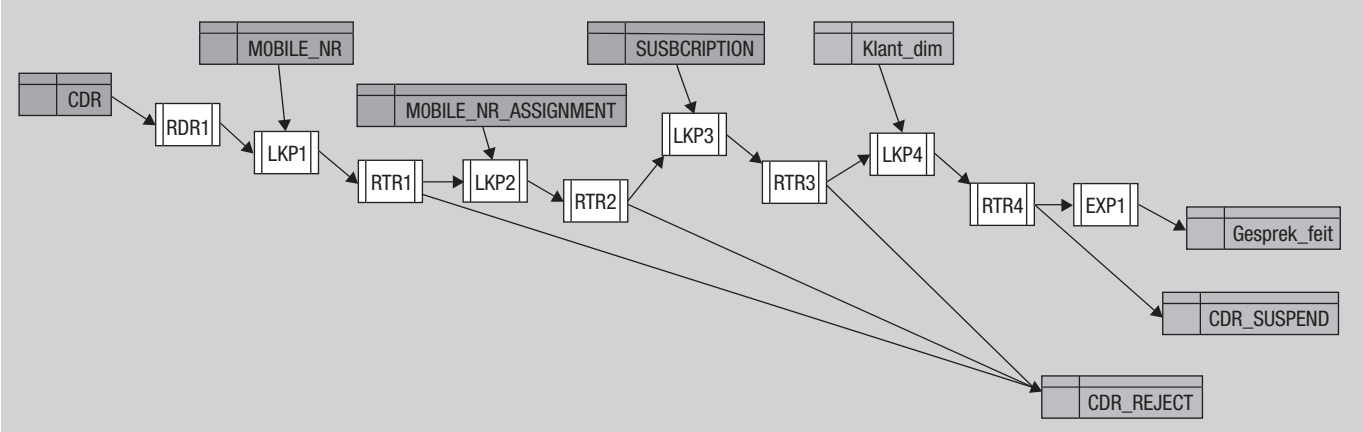
Regel	Stap	Criterium
1	Ophalen	CDR
2	Verrijken	CDR.MOBILE_NUMBER = MOBILE_NUMBER.MOBILE_NUMBER
3	Verrijken	CDR.MOBILE_NUMBER = MOBILE_NUMBER_ASSIGNMENT.MOBILE_NUMBER -en- CDR.STARTDATETIME tussen MOBILE_NUMBER_ASSIGNMENT.VALID_FROM en MOBILE_NUMBER_ASSIGNMENT.VALID_UNTIL
4	Verrijken	MOBILE_NUMBER_ASSIGNMENT.SUBSCRIPTION_ID = SUBSCRIPTION.SUBSCRIPTION_ID
5	Selecteren	CDR.STARTDATETIME is gisteren
6	Selecteren	MOBILE_NUMBER.STATUS != 'TEST'
7	Koppelen	KLANT.KLANTNUMMER = SUBSCRIPTION.CUSTOMER_NR -en- CDR.STARTDATETIME tussen KLANT.GELDIG_VANAF en KLANT.GELDIG_TOT
8	Valideren	Verrijking met MOBILE_NUMBER is geslaagd Indien niet geslaagd dan afwijzen
9	Valideren	Verrijking met MOBILE_NUMBER_ASSIGNMENT is geslaagd Indien niet geslaagd dan afwijzen
10	Valideren	Verrijking met SUBSCRIPTION is geslaagd Indien niet geslaagd dan afwijzen
11	Valideren	Koppeling met KLANT-dimensie is geslaagd Indien niet geslaagd dan in herverwerking zetten
12	Converteren	GESPREKSDUUR = CDR.DURATION
13	Converteren	OMZET = CDR.CHARGE

Afbeelding 5: Transformatieregels gespreksfeit.

LKP4 zoekt het juiste KLANT-dimensierecord op (koppelen#7). RTR4 routeert de niet geslaagde koppelingen naar de suspend-tabel, van waaruit eventueel herverwerking kan plaatsvinden (validatie#11).
EXP1 is de 'expression evaluator' die de gespreksduur en de omzet berekent (converteren #12 en #13).
Een bruikbaar hulpmiddel om te controleren of de gebouwde mapping de volledige beschreven functionaliteit dekt, is een kruistabel. Deze kruistabel legt een visueel verband tussen de functionele regels en de technische componenten, zie afbeelding 7.

Een vinkje in een cel geeft aan dat de technische component in die kolom de functionele regel van die rij realiseert (of een deel daarvan). In iedere rij of kolom moet dus ten minste 1 vinkje staan, anders is er óf een functionele regel niet gerealiseerd (rij leeg) óf er is een technische component die overbodig is.

Uitzonderingen hierop zijn uiteraard mogelijk, het is wel belangrijk deze dan als zodanig te beschrijven. Tevens geeft de kruistabel inzicht of wellicht een bepaalde logische stap dubbel is geïmplementeerd. Indien er bij een logische stap in méér dan



Afbeelding 6: Mapping diagram.

Regel	Stap	RDR1	LKP1	RTR1	LKP2	RTR2	LKP3	RTR3	LKP4	RTR4	EXP1
1	Ophalen	✓									
2	Verrijken		✓								
3	Verrijken				✓						
4	Verrijken						✓				
5	Selecteren	✓									
6	Selecteren			✓							
7	Koppelen								✓		
8	Valideren			✓							
9	Valideren					✓					
10	Valideren							✓			
11	Valideren									✓	
12	Converteren										✓
13	Converteren										✓

Afbeelding 7: Kruistabel Logisch-Technisch.

1 kolom een vinkje staat, is het raadzaam te onderzoeken of er niet sprake is van dubbel werk in de technische mapping.

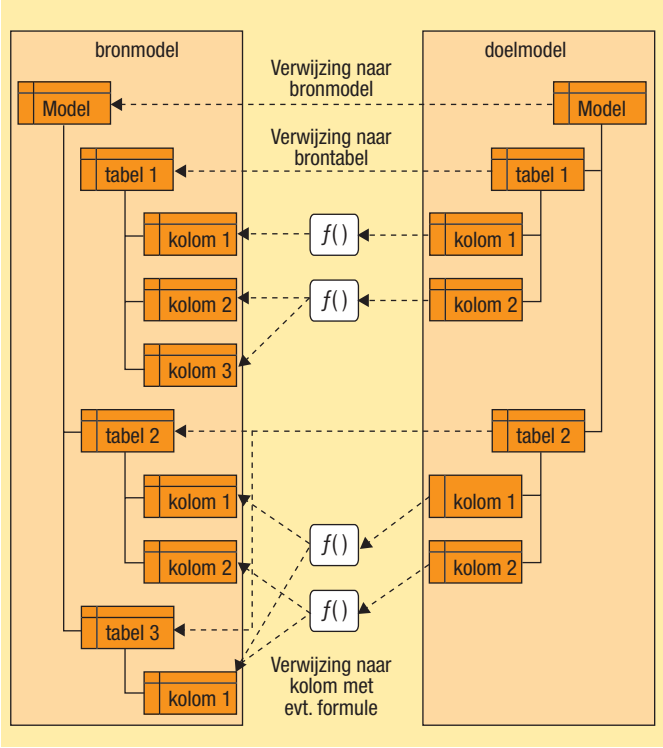
Case-tool en code-generatie

De vraag die zich nu voordoet is of dit ontwerp- en ontwikkelproces niet verregaande te automatiseren is. Met andere woorden, kan hier gebruik gemaakt worden van een case-tool met code-generatie? En bestaan er momenteel dergelijke tools op de markt? Ten eerste kunnen we constateren dat de designer-modules van de meeste ETL-producten zich al gedragen als case-tool en code-generator. Je schrijft als ETL-ontwikkelaar momenteel geen COBOL- of C-programma's meer, maar je modelleert de ETL in een grafische dataflow, zoals te zien in afbeelding 6. De ETL-tool vertaalt deze grafische dataflow vervolgens in programma-instructies. Het startpunt echter van deze tool-specifieke grafische modellering is (vanzelfsprekend) tool-specifiek, en dus niet platform-onafhankelijk. Het is (vooralsnog) niet mogelijk om bijvoorbeeld met de Designer-applicatie van Oracle Warehouse Builder een mapping grafisch te ontwerpen om hieruit vervolgens code voor Informatica PowerCenter te laten genereren. Gezien vanuit het belang van de verschillende productleveranciers ligt dit ook niet voor de hand.
De ETL-modellering die in dit artikel wordt beschreven ligt een abstractieniveau hoger dan dat van de verschillende ETL-tools, namelijk op het logisch niveau, en voor tool-ondersteuning dienen we daarom elders te zoeken.

ERM-tools

Datamodelleer-tools als Erwin en PowerDesigner bieden in de meest recente versies al enige ondersteuning voor het logisch 'mappen' van bron- en doel-entiteiten. Deze ondersteuning kan globaal als volgt worden beschreven. Bij het datamodel dat als 'doelmodel' wordt aangemerkt, wordt een verwijzing naar een ander datamodel gemaakt, dat dan als 'bronmodel' wordt aangemerkt. Aan één doelmodel kunnen meerdere bronmodellen gekoppeld worden. Vervolgens kan aan een tabel in het doelmodel een verwijzing naar één of meer tabellen van één van de betreffende bronmodellen worden gekoppeld. Weer een stap dieper kunnen aan een attribuut van de doeltabel verwijzingen worden gekoppeld naar attributen van de gekoppelde brontabellen. Bij een koppeling tussen doel-attribuut en bron-attributen is het mogelijk om tot op zekere hoogte een specificatie vast te leggen van de uit te voeren transformatie. In afbeelding 8 is dit schematisch weergegeven.

De geboden functionaliteit komt nog het meest overeen met de transformatiestap 'converteren' van het in dit artikel behandelde logische ETL-model. De mogelijkheden zijn echter beperkt. Voor het vastleggen van regels en criteria met betrekking tot de overige transformatiestappen bieden deze tools zeer geringe ondersteuning. De stap 'verrijken' wordt impliciet ondersteund door de 'foreign key' relaties tussen de tabellen in de bronmodellen. Deze relaties bestaan niet tussen tabellen uit verschillende bronsystemen, en daar houdt de ondersteuning dus direct op.
In PowerDesigner kan meer dan 1 mapping voor een doeltabel gespecificeerd worden, bijvoorbeeld voor het geval dat records uit twee bronnen verenigd worden in één doeltabel. De tool Powerdesigner is ook in staat om op basis van een attribuut-mapping een SQL SELECT-statement te genereren. Deze functionaliteiten biedt Erwin niet. De conclusie van de auteur is dat de ondersteuning van beide tools voor datamodel-mapping momenteel nog onvoldoende is voor een ETL-ontwerp. Wel kan deze gebruikt worden om een 'high-level' herkomstschema op te stellen tijdens de analysefase van een project.
Overigens komt de door deze tools geboden functionaliteit vrij veel overeen met het idee van de 'Logical Data Map', zoals dit door Ralph Kimball in zijn boek 'The Data Warehouse ETL Toolkit' wordt beschreven (pagina 58 en verder). Ralph Kimball gaf (in een gesprek met de auteur) aan dat de Logical Data Map inderdaad relatief oppervlakkig in het boek is behandeld, en gaf aan



Afbeelding 8: Data mapping in ERM-tools.

dat de methode van dit artikel een duidelijke stap in de goede richting is.

CWM-tools

Meer en meer datawarehouse-tools zijn CWM-compliant, wat staat voor Common Warehouse Metadata. In hoeverre is het CWM-model geschikt voor het logisch ontwerpen van ETL? En als dit al mogelijk is, in hoeverre kunnen ETL-tools dan een dergelijk model lezen en vertalen naar ETL-code?
Een rondje langs de websites van een aantal ETL-leveranciers (Informatica, Oracle, SAS, Microsoft, Ascential, Business Objects) levert het beeld op dat CWM vooralsnog wordt gezien als uitwisselings-standaard van metadata *nadat* de ETL-programma's ontwikkeld zijn. Vanuit de ontwerp-repository's kunnen gegevensmodellen en lineage-gegevens in CWM-formaat worden geëxporteerd en geïmporteerd. Verschillende metadata-management tools (bijvoorbeeld Informatica's Superglue) kunnen deze CWM-data integreren en hierover zeer uitgebreid rapporteren. Echter, deze CWM-data ontstaan *uit* het technische ETL-ontwerp, en dus niet andersom. De auteur heeft nog geen voorbeelden of artikelen kunnen vinden over de mogelijkheid om in 'CWM-taal' een transformatie logisch te specificeren, zodanig dat deze vervolgens ondubbelzinnig in ETL-code kan worden omgezet. Hij roept de lezers op om ervaringen op dit gebied met hem te delen. Aangezien de CASE-tools op dit moment geen soelaas bieden, is het gebruik van een sjabloon voor het opstellen van een logisch ETL-ontwerp, en een 'kookboek' voor het vertalen van dit ontwerp naar een tool-specifieke implementatie, de best bruikbare aanpak.

Conclusie

De methode voor logisch, i.e. platform-onafhankelijk, ontwerp van ETL-functies is op een gestructureerde manier te gebruiken bij het doorvertalen naar de techniek. Een tool-specifiek kookboek kan hierbij voor de gewenste uniformiteit en productiviteit zorgen. De ondersteuning voor geautomatiseerde ETL-generatie door CASE-tools is nog maar zeer beperkt ontwikkeld. Zowel de klassieke ERM-tools als het moderne CWM schieten hierin nu nog tekort. De ontwikkeling staat echter niet stil. Met name aan de kant van de ERM-tools gaan de ontwikkelingen meer in de richting van het ondersteunen van ETL-functies. Dat de hier beschreven ontwerpaanpak ook heel bruikbaar is voor het testen van ETL-programma's, zal in een volgend artikel worden uitgewerkt.

Mark Zwijsen
Mark Zwijsen (mark.zwijsen@atosorigin.com) is Senior Consultant Datawarehousing bij Atos Origin.

Update

FileMaker Server 8 beschikbaar

De nieuwe serversoftware voor efficiënt samengebruik van database-oplossingen is nu sneller in opdrachtverwerking. FileMaker Server 8 bedient maximaal 250 gelijktijdige gebruikers met 125 FileMaker-databasebestanden. FileMaker Server 8 heeft een verbeterde 'engine' en kan snel worden geïnstalleerd. De jongste telg biedt een verbetering van gedeelde databases, een betere gegevensbescherming dankzij verbeterde beveiligingsmogelijkheden en tijdswinst door uitgebreide beheermogelijkheden. Van de nieuwe FileMaker 8-producten zijn inmiddels FileMaker Server 8, FileMaker Pro 8 en FileMaker Pro 8 Advanced beschikbaar.
Meer informatie op www.filemaker.nl

Cognos introduceert Cognos 8 BI

Cognos, leverancier voor Business Intelligence en Corporate Performance Management, brengt Cognos 8 Business Intelligence op de markt. Dit is het eerste product dat complete BI-functionaliteit biedt op basis van één bewezen, op web-services gebaseerde architectuur, die werd geïntroduceerd met Cognos ReportNet in 2003.
Het heeft een eenvoudige, zero-footprint interface voor alle soorten gebruikers, rapportontwerpers en beheerders en biedt een simpele omgeving die acceptatie door gebruikers stimuleert, besluitvorming verbetert en bedrijfsbreed als technologiebasis kan dienen voor Performance Management.
Meer informatie op www.cognos.nl