De bouw van een metadata-driven, open source datawarehouse

Van spaghetti naar lasagne

Johannes van den Bosch

Als waterpartner in Midden-Brabant werkt Waterschap De Dommel aan voldoende, schoon en veilig water. Er wordt projectmatig gewerkt en via een Planning & Control (P&C) cyclus wordt gestuurd op geld en op uren. Dit vraagt om een heldere, geïntegreerde managementinformatievoorziening. Om ook in de toekomst te kunnen voorzien in de complexer wordende informatiebehoefte, is gekozen voor de bouw van een Enterprise Data Warehouse (EDW) op basis van Data Vault modeling.

Dit EDW wordt gerealiseerd met open source software en wordt grotendeels gegenereerd in plaats van met de hand gebouwd. Zo kon zonder begininvestering een vliegende start gemaakt worden met datawarehousing.

We schrijven 2004. Om geïntegreerd en centraal te kunnen rapporteren over twee financiële systemen wordt BusinessObjects aangeschaft, toen nog 'rapportgenerator' genoemd in de projectdocumentatie. Daarnaast wordt een Business Intelligence Competence Center (BICC) geïnstalleerd.

De BI-architectuur is eenvoudig van opzet: er is sprake van een centrale portalomgeving (InfoView) waarin rapportages aangeboden worden; de bronsystemen zijn ontsloten via een *universe* (een metadatalaag van BusinessObjects waarin business logica vastgelegd wordt zoals relaties, filters en naamgevingen) en op deze *universes* worden rapportages gebouwd.

Kortom, er wordt direct op de bron gerapporteerd. De rapportages zijn beperkt systeemoverstijgend, zodat er een lage mate van integratie is en er zijn geen andere ontsluitingsvormen dan rapportages. Dit bleek een prima basis voor de jaren erna. Het BICC werd een succes. Er werd echter een punt bereikt waarbij het aanbod niet meer kon aansluiten bij de complexer wordende vraag.

Uitdagingen

Het BICC werd geconfronteerd met een aantal uitdagingen, waardoor tegen de beperkingen van de huidige BI-omgeving werd aangelopen.

1. Informatiebehoefte complexer.

Hoewel het informatieaanbod beperkt systeemoverschrijdend is, wordt de vraag naar geïntegreerde informatie steeds groter. Een voorbeeld: uren dienen gekapitaliseerd te worden. Dat wil zeggen dat uren, omgerekend naar euro's, direct op het budget van een projectleider gaan drukken. Het is dus van belang om zowel op geld als op tijd te sturen. Waar voorheen projectmanagement en tijdmanagement nog twee afzonderlijke activiteiten waren (ondersteund door afzonderlijke systemen), dient de informatie nu geïntegreerd geleverd te worden als stuurinformatie.

2. Meer ontsluitingsvormen en leveranciersafhankelijkheid.

Omdat rapportages al snel lijken op verantwoordingsinformatie (wat is er gebeurd?), ontstond een behoefte aan stuurinformatie (waarom is het gebeurd?). Dit is een natuurlijke ontwikkeling die ook terug te vinden is in diverse BI-maturitymodellen. Zo werd er gevraagd om management dashboards en analytics voor power-users.

Voor nieuwe ontsluitingsvormen zijn we echter afhankelijk van onze huidige leverancier. De inmiddels zeer complex geworden spaghetti die *universe* heet, bevat onze business logica; wil je deze hergebruiken, dan ben je afhankelijk van software van dezelfde leverancier. Een klassiek gevalletje van vendor lock-in. Om dit in de toekomst te voorkomen willen we onafhankelijker worden van leveranciers. Zo willen we kunnen kiezen voor de beste tools per geval en niet gebonden zijn aan het aanbod van één leverancier. We willen kunnen switchen naar andere oplossingen zonder omvangrijke herinvesteringen. Dit vereist een gelaagde architectuur waarin bepaalde problemen (zoals het vastleggen van historie, het integreren van gegevens of het toepassen van businesslogica) centraal worden opgelost.

3. Professionaliteit.

De behoefte ontstaat om business logica en andere metadata professioneler te gaan beheren. De broneigenaar is eigenaar van de data, de business is eigenaar van de business logica en het BICC is eigenaar van het proces dat leidt tot informatie. Dit moet door middel van traceerbaarheid aan te tonen zijn. De BI-architectuur wordt daardoor auditeerbaar.

4. Flexibiliteit.

Er wordt gestreefd naar lage beheerlasten voor de BI-omgeving; ook in de toekomst waarin er steeds meer bronnen worden ont-

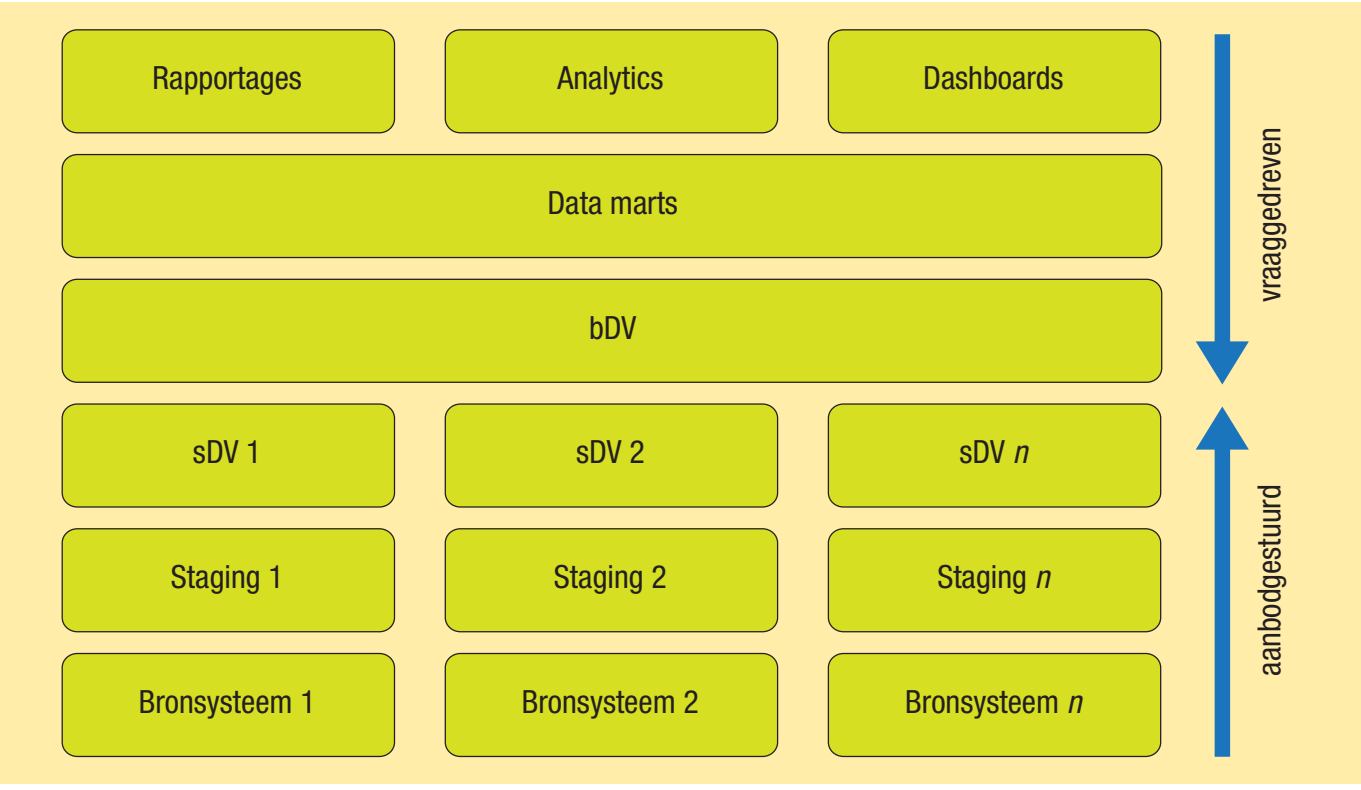
sloten en de informatiebehoefte groeit. Het moet mogelijk zijn om veranderingen (zoals het wijzigen van business logica of het toevoegen van systemen) gecontroleerd en snel door te voeren met een lage impact. Er moet lineaire uitbreidbaarheid worden nagestreefd in plaats van exponentieel stijgende beheerlasten te ervaren.

5. Service Oriented Data en Master Data Management.
Met ontwikkelingen als de OverheidsDataBase (ODB) lijken servicegeoriënteerde en real-time data op de waterschappen af te komen. Het plan is om gemeenschappelijke basisgegevens via services aan te gaan bieden. Dit wordt een uitdaging, niet alleen op BI-gebied maar ook qua gegevensbeheer. Deze gegevensstroom zal moeten worden opgevangen, bewaard, en worden gedistribueerd naar bestaande systemen. Activiteiten onder de noemer masterdata management (MDM) moeten worden ontplooid. Onze visie is dat dit met eenzelfde fundament verwezenlijkt moet kunnen worden als het fundament van een BI-omgeving. Wij zien dit niet als twee losstaande problemen.

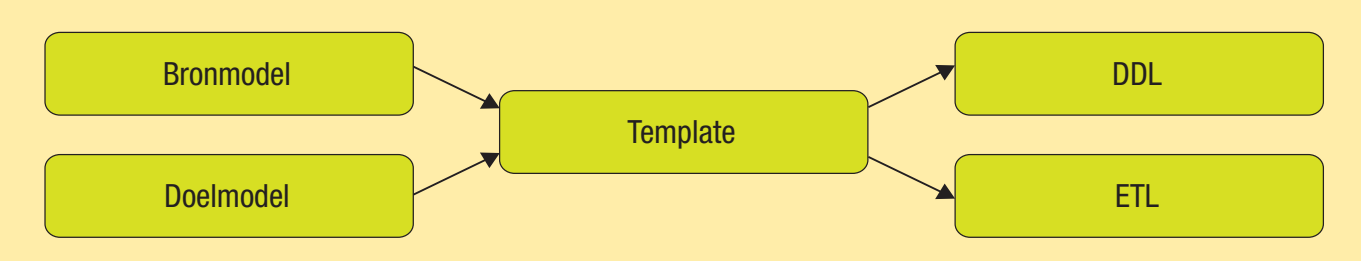
Architectuur

Om deze uitdagingen aan te kunnen is een nieuwe BI-architectuur ontworpen. Deze architectuur kenmerkt zich door gelaagdheid, een hoge graad van automatisering en het toepassen van open source en open standaarden. De gelaagdheid is vrij naar het Corporate Information Factory model van Inmon [1] ontworpen.
De lagen zijn in te delen in data-acquisitie (bestaande uit bronsystemen en staging area's), databeheer (source Data Vaults en een business Data Vault) en datalevering (datamarts en bijbehorende front-end tools). Zoals te zien in de schematische weergave in afbeelding 1, kent deze op een lasagne lijkende architectuur de volgende lagen:

- Bronsystemen zijn de ontsloten operationele systemen. Dit kunnen databases zijn van applicaties, maar ook minder gestructureerde data of real-time data op basis van services;
- Een staging area zorgt voor gemak van verdere verwerking van de data, data worden batchgewijs een-op-een uit de verschillende bronsystemen op een uniforme manier en locatie opgeslagen en vanaf daar door de volgende laag opgepikt. De staging area wordt gegenereerd, zowel qua structuur als qua laadroutines (ETL);
- De source Data Vault (sDV) draagt zorg voor historievastlegging. Het betreft een volgens Data Vault Modeling [2] gemodelleerde laag waarbij het bronmodel leidend is. Het bronmodel uit de staging area wordt omgezet in een Data Vault model. Elk bronsysteem heeft zijn eigen sDV. De sDV houdt historie vast, er worden alleen maar data toegevoegd. Wordt er dus een wijziging gedetecteerd in het bronsysteem, dan wordt het bestaande record in de sDV afgesloten en een nieuw record aangemaakt (start geldigheid datum). Worden data verwijderd, dan wordt het record in de sDV afgesloten. In de sDV worden geen transformaties uitgevoerd. Cleansing en integratie vindt bewust later, benedenstrooms plaats. Hierdoor is traceerbaarheid en auditeerbaarheid gewaarborgd. De staat van een bronsysteem is immers voor een willekeurig peilmoment te reconstrueren vanuit de sDV. De sDV wordt



Afbeelding 1: Schematisch overzicht



Afbeelding 2: Werking Quipu.

gegenereerd, zowel qua structuur als qua laadroutines. De bouw van de sDV is dus aanbodgedreven;

- Bovenop de verschillende sDV's bevindt zich één business Data Vault (bDV). Het doel van deze laag is het toepassen van herhaalbare business logica. Deze wordt gemodelleerd met een 'businessbril'. Als eerste zorgt deze laag voor de vertaling naar een bedrijfsgegevensmodel. Het bDV model wordt volgens de business ontworpen. Entiteiten krijgen herkenbare namen (bijvoorbeeld costcntr_hub in de sDV wordt kostenplaats_hub in de bDV). Alleen elementen die gebruikt worden volgens de business worden gemodelleerd. De bouw van de bDV is dus vraaggestuurd.
  Ten tweede zorgt deze laag voor business key integratie. Dit houdt in dat entiteiten die in verschillende systemen geregistreerd worden (in de praktijk zichtbaar als unieke business keys) in één bDV-entiteit geïntegreerd worden. Een voorbeeld: als medewerkers in systeem A (entiteit 'employee') en systeem B (entiteit 'person') geregistreerd worden, worden de bijbehorende sDV entiteiten in de bDV samengevoegd als één entiteit (entiteit 'medewerker') door het toepassen van business logica (bijvoorbeeld: employeenumber in systeem A moet gelijk zijn aan personcode in systeem B).
  De bDV is geïmplementeerd als een collectie views. De bDV ligt dus als het ware als een semantische laag bovenop de sDV's. Er wordt niet direct gerapporteerd op de sDV of bDV. Gebruikers hebben geen toegang tot deze lagen. Hiervoor worden datamarts ontworpen;
- De datamarts worden waar mogelijk gebouwd als views bovenop de bDV. Het creëren van deze views is eenvoudig. De datamarts (DM's) bestaan uit een stermodel met feiten en dimensies. Een feitentabel wordt opgebouwd uit een bDV-link (plus bijbehorende satelliet), een dimensie wordt opgebouwd uit een bDV-hub (plus bijbehorende satelliet). Heeft de eindgebruiker de behoefte aan een *slowly changing dimension* (SCD), dan wordt historie uit de satelliet meegenomen, zo niet, dan worden alleen actuele records geselecteerd. Historie wordt immers al in de sDV vastgelegd en herhaalbare business logica is reeds toegepast in de bDV.
  DM's zijn op deze wijze niet 'duur' om te maken en zijn dus *disposable*, wegwerpbaar. Een DM is puur een afleiding van de bDV, kan snel worden gecreëerd en kan dus ook weer snel weggegooid worden als er geen noodzaak meer voor is. Er gaan geen data verloren, deze zitten immers in de Data Vault.
  Desgewenst kan er specifieke business logica worden toegepast in deze DM, bijvoorbeeld wanneer er gevraagd wordt om specifieke filtering. Hierdoor zijn we niet gebonden aan een 'single version of the truth', die in de praktijk veelal niet bestaat. Ook kunnen DM's met behulp van ETL-tooling gerealiseerd worden, bijvoorbeeld wanneer de complexiteit de mogelijkheden van views overstijgt. Denk aan aggregaties of complexe rekenregels;
- Front-end tools worden gebruikt door de eindgebruiker. De tools halen hun data uit de zeer gestructureerde DM's. Deze tools kunnen bestaan uit rapportages, maar ook dashboards, analyse (OLAP) en datamining-toepassingen.

Automatiseren waar mogelijk

Het EDW kenmerkt zich door een hoge mate van automatisering; handwerk kost immers tijd en is foutgevoelig. Een belangrijke eigenschap van een Data Vault model is herhaalbaarheid. De patronen van een Data Vault (entiteiten, afleidingsregels en laadroutines) zijn herhaalbaar en dus te automatiseren. Zeker in het geval van een sDV. Bronmodellen worden *reverse engineered* en omgezet naar een Data Vault model; zowel de structuur als de laadroutines van de sDV worden volledig gegenereerd. Slechts in het geval dat het bronsysteem geen keys en/of relaties heeft gedefinieerd (*foreign key constraints*) is er een eenmalige handmatige actie nodig (het aangeven van de sleutelvelden en definiëren van de relaties).
De laadroutines (ETL) en structuur (DDL) van de sDV bestaan uit SQL-statements (ANSI SQL compatible) en zijn daarmee dus database-onafhankelijk. Deze SQL wordt gegenereerd aan de hand van templates. Deze templates kunnen indien gewenst ook worden aangepast.
Ook de staging area is logisch en herhaalbaar en wordt dus gegenereerd. De structuur (DDL) betreft ook weer ANSI SQL (CREATE TABLE statements). De benodigde laadroutines (ETL) worden ook gegenereerd, in het specifieke formaat van onze data-integratietool (zie Tools en techniek).

Handmatig waar nodig

De bDV wordt handmatig gemaakt. In onze architectuur wordt de bDV niet fysiek geïmplementeerd, dat wil zeggen dat de bDV in onze eerste opzet gebouwd is als een verzameling views (wederom SQL) bovenop de sDV's. Ook de DM's zijn in de eerste opzet views.

Deze views zijn eenvoudig van opzet. Alle complexe handelingen zoals het vasthouden van historie zijn immers al in onderliggende lagen opgelost, waardoor handwerk voor ons geen bezwaar is. De bouw van de bDV en de DM's verloopt daardoor snel. Alle business logica is nu gecentraliseerd, wordt centraal gedocumenteerd in een Wiki en wordt afgestemd met de business.
Een entiteit in een DM haalt dus gegevens uit de business Data Vault (bDV) die weer gegevens uit de source Data Vault (sDV) haalt. Het voordeel is dat de sDV op een willekeurig moment bijgewerkt kan worden, zonder verstoringen benedenstrooms. We zijn dus niet afhankelijk van batchverwerkingen die alleen 's nachts kunnen plaatsvinden.
Een nadeel zou performance kunnen zijn. Dit ervaren wij niet, onder andere omdat de views gebruik maken van de onderliggende indexen en er momenteel geen gigantische hoeveelheid data verwerkt wordt. Mocht de performance een probleem worden, dan kan een view vrij eenvoudig fysiek gemaakt worden (*materialized views*). Ook zou hiervoor een ETL-tool ingezet kunnen worden. De gelaagde architectuur laat deze flexibiliteit toe.

Tools en techniek

Het genereren van staging en source Data Vault gebeurt met behulp van een open source datawarehouse-managementtool: Quipu [3].
Uitvoeren van de ETL gebeurt met de open source data-integratietool Pentaho Data Integration (PDI, voorheen Kettle) [4]. Beide tools hebben wij geïntegreerd; zo wordt de door Quipu gegenereerde SQL door PDI uitgevoerd en gemonitord. Ook hebben we een template ontwikkeld die staging-ETL genereert voor PDI, die deze ETL uitvoert. Deze zaken stellen wij, volgens de open source gedachte, beschikbaar aan de community.
Daarmee hebben we een duidelijke scheiding aangebracht: Quipu wordt gebruikt voor het beheer van het datawarehouse (genereren) en PDI om daadwerkelijk data te laden en te transformeren (uitvoeren). Beide tools doen dus waar ze voor gemaakt zijn, maar zijn wel gekoppeld. Daardoor ontbreekt ook hier (dubbel) handwerk.

Samenvattend

In het begin van dit artikel werd een aantal uitdagingen genoemd. Om te beginnen de behoefte aan geïntegreerde managementinformatie. Het leveren hiervan wordt eenvoudig mogelijk: integratie wordt namelijk al verzorgd door één van de lagen (de bDV). De datamarts waarop gerapporteerd wordt zijn daardoor vanzelf systeemoverstijgend.
Ook het vasthouden van historie en het toepassen van business logica (integratie, transformatie) is centraal opgelost in het EDW. De uiteindelijke DM's zijn zeer leesbaar en eenvoudig van opzet. Hierdoor kan met willekeurige rapportagetools gewerkt worden. Ook andere tools, zoals dashboards en interactieve analyse, kunnen eenvoudig aansluiten op deze DM's. De afhankelijk van de producten van één leverancier is daarmee dus verdwenen.
De Data Vault aanpak biedt auditeerbaarheid en traceerbaarheid en business logica krijgt een rechtmatige eigenaar: de business. Een belangrijke eigenschap van een Data Vault model is dat toevoegingen géén impact hebben op bestaande fysieke Data Vault structuren, er ontstaan alleen nieuwe Data Vault elementen; bestaande elementen veranderen niet. Met andere woorden, de impact van uitbreidingen is duidelijk te overzien: komt er (een deel van) een bronsysteem bij, dan genereren we eenvoudigweg de bijbehorende staging en de sDV. Daarna wordt de bDV uitgebreid. Bestaande DM's merken hier niets van, er is benedenstrooms geen impact.
Omdat de business logica centraal en eenmalig toegepast wordt (bDV), zijn ook wijzigingen in business logica eenvoudig door te voeren. Dit in tegenstelling tot een traditioneel datawarehouse waarin business logica en historie verweven zijn. Kortom, het EDW wordt lineair uitbreidbaar. Daardoor kan desgewenst klein worden begonnen.
Een Data Vault is een goede basis om servicegeoriënteerde en real-time data vast te leggen. Berichtenverkeer kan getransformeerd worden naar een Data Vault model; binnenkomende data worden vastgelegd, inclusief historie, en transacties hoeven niet volledig te zijn om geladen te kunnen worden.

> Het EDW wordt lineair uitbreidbaar

Een binnenkomende service of bus krijgt zijn eigen sDV. Dit is de rauwe 'fact store'. Deze data worden omgezet in een business model (bDV) en geïntegreerd met bestaande bronnen, daarbij rekening houdend met afspraken over welke systemen leidend zijn. Vanaf daar kunnen eenvoudig data worden gedistribueerd naar bestaande systemen (ETL). Dit kan allemaal in onze bestaande BI-architectuur met de bestaande tools. We hoeven dus geen aparte activiteiten op te starten.
Het allergrootste voordeel van de aanpak is voor ons dat er zonder investering een vliegende start gemaakt kan worden met de bouw van een EDW. Er hoeft niet vooraf geïnvesteerd te worden in licenties, en door de hoge graad van automatisering kan snel naar resultaat toe worden gewerkt. Zijn er nu nog redenen om *niet* voor een Data Vault gebaseerd EDW te kiezen?

Literatuur

1. *Corporate Information Factory (CIF), Bill Inmon, www.inmoncif.com/home/*
2. *Data Vault Modeling, Dan Linstedt, http://danlinstedt.com/about/data-vault-basics/*
3. *Quipu, www.datawarehousemanagement.org/*
4. *Pentaho Data Integration, http://kettle.pentaho.com/*

Johannes van den Bosch

Drs. J.M. van den Bosch (jvdbosch@dommel.nl) is BI-specialist bij Waterschap De Dommel.