Een strategie om problemen bij implementaties te ondervangen

Testen van datawarehouses

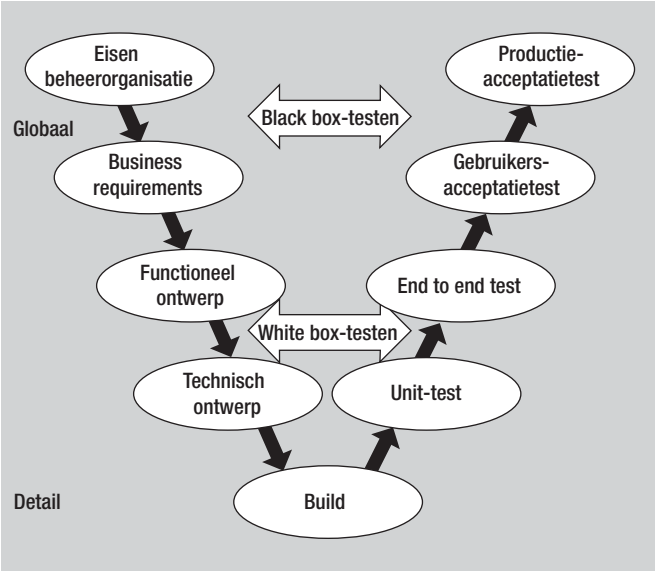
Tom van Maanen en Eli Diemer

Bij het testen van een datawarehouse wordt men geconfronteerd met twee voor een datawarehouse-traject typische probleemvelden. Ten eerste gaat het om erg veel data die verwerkt worden, ten tweede zijn er veel programma's die zorgen voor extractie en opslag van data. In dit artikel wordt een strategie beschreven om met deze problemen om te gaan.

De strategie is geënt op de traditionele testmethode (V-model). In feite wordt een concrete invulling gegeven van het V-model in de context van datawarehouse-implementaties in de praktijk. Testen van een datawarehouse wordt als complex ervaren. Veel testers zijn vertrouwd met het testen van online applicaties en staan vreemd tegenover een datawarehouse dat er heel anders uitziet.

Drie halteplaatsen

Voor de tester zijn er twee probleemvelden. Het eerste probleemveld betreft de grote aantallen records. Dit vloeit voort uit de aard van een datawarehouse, waarbij data van operationele systemen worden gekopieerd naar een datawarehouse-omgeving om zodoende een eigen omgeving te hebben die is toegerust voor rapportage- en analysedoelen. De operationele systemen leveren veelal duizenden tot miljoenen records per dag op die verwerkt worden in een datawarehouse. In samenhang daarmee duren de betreffende query's ook lang; een doorlooptijd van een nacht is dan geen uitzondering. Een tester die geen rekening houdt met dergelijke grote aantallen records en met de lange doorlooptijden verdrinkt dan als het ware in de datamassa. De teststrategie moet beslist rekening houden met de hoeveelheid en zal expliciet in moeten gaan op de risico's die hiermee samenhangen.
Het tweede probleemveld betreft de grote aantallen programma's die worden gebruikt in een datawarehouse. Men wil in een datawarehouse immers graag data uit verschillende operationele systemen kopiëren. Als er dan per operationeel systeem verschillende databronnen zijn, zijn al direct enige tientallen datastromen te zien.In de meeste datawarehouse-omgevingen wordt onderscheid gemaakt naar: 1. een operational datastore om de invoerdata op te vangen; 2. een datawarehouse om de data in op te slaan; en 3. datamarts die aan analisten ter beschikking worden gesteld.



Afbeelding 1: Het gevolgde proces bij de bouw van een datawarehouse.

Dat betekent dat de datastromen ten minste drie halteplaatsen hebben. Dit leidt dan al weer snel tot tientallen, zo niet honderden, programma's die een rol spelen om de gegevens op juiste wijze door een datawarehouse te leiden. Deze programma's zijn ook nog eens onderling gerelateerd: de uitvoer van het ene programma is mogelijk de invoer voor een volgend programma. In de teststrategie moet hier dan ook weer rekening mee gehouden worden.

V-Model

Er is behoefte aan een goede teststrategie om het complexe geheel van een datawarehouse aan te vatten. De strategie, geënt op het traditionele V model, wordt op een zodanige manier gebruikt dat men de eventuele problemen van het testen van een dataware-

house zo handig mogelijk kan aanpakken. Het bouwen van een datawarehouse is een project dat zich afspeelt in een bestaande organisatie die bepaalde eisen kent. Het datawarehouse-project begint bij het vastleggen van de *business requirements*. Van daaruit wordt tot in detailniveau een functioneel ontwerp gemaakt, dat dan weer wordt uitgewerkt in een technisch ontwerp, waarna het uiteindelijk gebouwd wordt. De richting is daarbij steeds van generalistisch naar detailistisch, zie afbeelding 1.
Het testen volgt de omgekeerde route: er wordt gestart op detailniveau. Als eenmaal is vastgesteld dat de programmatuur goed functioneert op detailniveau, worden complexere niveaus getest. Dit gaat door totdat het gehele systeem kan worden getest en op orde wordt bevonden. Bij de uiteindelijke acceptatie wordt gecontroleerd of het systeem onder operationele omstandigheden aan de volledige set van requirements voldoet.

Unit-test

De eerste serie van testen zijn de unit-testen, waarin alle afzonderlijke programma's één voor één worden getest. Deze testen worden gedaan met een beperkt aantal invoer-records, waarbij de attributen representatief zijn voor de domeinen die in de operationele systemen gebruikt worden. Het verschil is gelegen in de aantallen invoer-records van de programma's. Dit soort testen doet men met hooguit enkele tientallen records. Het uiteindelijke systeem zal met duizenden, zo niet miljoenen records geconfronteerd worden.

> Voor ontwikkelaars is de test een mooie gelegenheid om de performance te verbeteren

In eerste instantie wil men zeker weten dat de programma's inhoudelijk juist werken en in functionele zin correct zijn. Aan de hand van de rekenregels uit het ontwerp is het mogelijk om bij een invoer van enkele tientallen records de uitvoer exact te voorspellen. De unit-test wordt gedaan met een beperkte testset en daardoor is per testgeval tevoren de vereiste uitvoer te voorspellen. Deze voorspelling wordt vergeleken met de uitvoer van het programma. De vergelijking van voorspelde uitvoer en feitelijke uitvoer is ook weer een overzichtelijke exercitie: in de meeste gevallen zal het om tabellen van hooguit enkele tientallen records gaan. Verschillen tussen voorspelde uitkomst en feitelijke uitkomst kunnen snel gevonden en relatief eenvoudig verklaard worden. Dankzij de beperkte invoer-set, is de doorlooptijd van de testen ook kort: er hoeft maar een beperkt aantal records verwerkt te worden.
De unit-testen zijn te typeren als *white box-testen*: in principe wordt iedere rekenregel uit het ontwerp getest en wordt gekeken of deze correct gecodeerd is. Bij dit soort testen kan men ook *code walkthrough* doorvoeren, waarbij nagegaan wordt of de code overeenkomt met de rekenregels uit het ontwerp.
Op zich leent deze test zich ook voor automatisering van het testen. In een script kan men de gewenste invoerbestanden gereed zetten voor verwerking, vervolgens kan men het betrokken programma laten draaien. Tot slot kan men in het script de verkregen uitvoer vergelijken met voorspelde uitvoer.

De unit-test richt zich op de juiste werking van ieder van de vele programma's die onderdeel vormen van het datawarehouse. Deze programma zijn één voor één gespecificeerd in het technisch ontwerp van het datawarehouse. De technisch ontwerpers zijn dan ook de aangewezen personen om deze unit-tests te ontwerpen. Dergelijke testen kan men na het schrijven van het technisch ontwerp opzetten, terwijl er gebouwd wordt aan de programma's. Als de programma's eenmaal klaar zijn, heeft men ook de beschikking over deze unit-tests, die dan ook weer direct kunnen worden afgenomen. Op deze manier kan de kennis die in een technisch ontwerp is verkregen ook goed worden toegepast in het ontwerp van de unit-tests.
De unit-test heeft een beperkte scope; een unit-test alleen voldoet niet. Diverse zaken moet men nog goed onderzoeken. Daaronder vallen dan testen waarbij de onderlinge afhankelijkheid van programma´s wordt onderzocht en de testen waarbij gewerkt wordt met grote aantallen records. Dit komt in volgende testen aan de orde.

End to end test

In de End to end test (E2E) worden de onderlinge afhankelijkheden onderzocht tussen programma's. Hierbij worden reeksen programma's getest, waarbij de uitvoer van het ene programma dient als invoer voor het andere programma. In een datawarehouse bestaat normaliter een fase waarin de invoer wordt opgevangen en bewerkt (de operational data store), vervolgens is er het datawarehouse zelf en een aantal datamarts ten behoeve van analyse door gebruikers, de eerder vermelde drie halteplaatsen voor de data. Dit betekent ook dat drie keer uitvoer van een programma gebruikt wordt als invoer voor een volgend programma. In de voorgestelde versie van de E2E test, wordt gewerkt met kleine invoertabellen. Dat betekent dat men in staat is om de uitvoer aan het eind exact te voorspellen op basis van de beperkte invoer aan het begin van de keten. Bij de E2E wordt gebruik gemaakt van het feit dat men weet dat de afzonderlijke programma's goed werken, dat is immers getest in de unit-tests. Op deze manier bouwt de E2E test logisch voort op de unittests. Ten opzichte van de eerdere tests hoeft men met de E2E test alleen nog te verifiëren of de programma's op een juiste wijze met elkaar samenwerken.
Hoe dit precies ingericht wordt hangt af van de opbouw van het datawarehouse. Er is een indeling te maken aan de hand van de betrokken operationele systemen. Per operationeel systeem wordt bekeken of de opgeleverde data van een operationeel systeem in de datawarehouse-omgeving goed verwerkt worden.

Per operationeel systeem realiseert men een op dat operationeel systeem georiënteerde testset.
Bij de E2E test kan ook de scheduler getest worden om de verschillende programma's in een keten achter elkaar te laten uitvoeren. De scheduler wordt daarbij zo ingericht dat afzonderlijke programma's pas van start gaan als voldaan is aan de begincondities van het betreffende programma. De scheduler zorgt ervoor dat eerst de extractieprogramma's worden getest die data uit operationele systemen verwerken. Vervolgens worden de programma's gestart die voor verdere verwerking van data zorgen.

De E2E test kan gedurende het testtraject steeds complexer worden gemaakt. Als iets eenmaal naar tevredenheid blijkt te werken, kan de test uitgebreid worden met meerdere datastromen. Dat gaat door tot het gehele datawarehouse met alle datastromen in de E2E test wordt gevat.
De E2E richt zich op een goede samenwerking van afzonderlijke programma's. Bij wijze van voorbeeld kan men denken aan een juiste verwerking van invoerbestanden uit een bepaald operationeel systeem. De werking van een dergelijke datastroom is beschreven in het functioneel ontwerp. Het is dan ook verstandig om de schrijvers van het functioneel ontwerp dergelijke functionele acceptatietesten te laten schrijven.

Voor de functioneel ontwerpers is een goed moment om deze testen te gaan maken na het schrijven van het functioneel ontwerp, ze kunnen zich dan wijden aan de functionele acceptatietest. Op deze manier wordt de kennis goed gebruikt in het testtraject. Zij zijn goed op de hoogte van de volgorde waarin de bewerkingen moeten plaats vinden en hebben dat in het functioneel ontwerp gezet, het is dan ook goed dat ze nagaan of het correct is verwerkt.
Met de E2E test is de onderlinge afhankelijkheid van de programma's in een datawarehouse getest. Daarmee weet men dat de programma's onderling goed samenwerken en dat de uitvoer van toeleverende programma's op de juiste wijze wordt gebruikt als invoer. Er is evenwel nog steeds gewerkt met kleine bestanden, op deze manier was men in staat om snel te werken. Een test is dan snel gedaan en fouten kunnen snel opgespoord en opgelost worden.
In deze fase wordt nog niet gewerkt met volledige bestanden zoals die in een productieomgeving gebruikt worden. Men weet dus nog niet of het systeem wel performant is.

Gebruikersacceptatietest

Met de gebruikersacceptatietest wordt getest met de volledige set van invoerbestanden. De opgeleverde datamarts zijn dan ook representatief voor wat de gebruiker gaat zien in het toekomstige datawarehouse. Omdat de datamarts zullen lijken op de uiteindelijke situatie, kan men deze testen overlaten aan de eindgebruikers. Om die reden is deze test dan ook de 'gebruikersacceptatietest' genoemd. De gebruikersacceptatietest bouwt voort op de eerdere tests, omdat van reeksen van programma's al is vastgesteld dat deze correct zijn. Ten opzichte van de eerdere tests moet nu alleen nog geverifieerd worden of de programma's ook met grote aantallen records kunnen omgaan – er is al vastgesteld dat er met kleine databestanden op de juiste wijze wordt omgegaan.
De gebruikersacceptatietest wordt bij voorkeur ontworpen door degenen die betrokken zijn geweest bij het opstellen van business requirements; de vanuit de 'business' gestelde eisen aan de uitkomsten van de datawarehouse-omgeving. Als er sprake is van op te leveren rapporten, weten de opstellers hoe deze rapporten beoordeeld moeten worden. Zo kunnen zij ook het beste de gebruikersacceptatietest opstellen, voor het gereed komen van het systeem is er voldoende tijd om een dergelijke test te ontwerpen.

> De technisch ontwerpers zijn de aangewezen personen om de unit-tests te ontwerpen

De gebruikersacceptatietest is geen eenvoudige opgave. In een ideale situatie moeten de gebruikers nagaan of ze de cijfers van de datamart kunnen herleiden tot berekeningen die gedaan zijn op de operationele systemen. Bijvoorbeeld: als er in de datamarts gesproken wordt over een omzet van een bedrag in euro's, dan zou dat zelfde bedrag in euro's ook terug te vinden moeten zijn in de operationele systemen, die de data hebben aangeleverd aan het datawarehouse. Dit is een manier van testen, die men als een black box kan betitelen: het systeem wordt niet op interne verwerking getest.
Voor de ontwikkelaars is deze test ook een mooie gelegenheid om de performance te verbeteren. In deze test is er immers voor de eerste keer sprake van een volledige invoer-set. Daarmee zullen ongetwijfeld knelpunten boven water komen. In deze fase is er dan gelegenheid om dit te verbeteren.

Als deze test succesvol is afgesloten, weet men dat er een goed werkend datawarehouse bestaat. Deze is ingericht en onderhouden door het ontwikkelteam. Op enig moment neemt het ontwikkelteam afscheid en moet het beheer worden overgedragen aan beheerteams. Of deze redelijkerwijs in staat zijn om dat beheer over te nemen is nog niet getest. Daarvoor dient een volgende test.

Productie-acceptatietest

De productie-acceptatietest (PAT) heeft als doel om de beheerorganisatie in staat te stellen na te gaan of deze de verantwoordelijkheid van beheer van het datawarehouse over kan nemen van het ontwikkelteam. De productie-acceptatietest kan parallel uitgevoerd worden aan de End-to-End test of aan de gebruikersacceptatietest. Bij deze test moet antwoord gegeven worden op een serie vragen:

- Is de programmatuur wel stabiel: met andere woorden kan het gedurende een afgesproken periode draaien zonder storing;
- Voldoet het opgeleverde systeem aan de door de afdeling productie gestelde eisen;
- Zijn de procedures in overeenstemming met de procedures die op de beheerafdeling gelden;
- Kan de accountantsdienst of de afdeling interne controle zich verenigen met de manier waarop met data wordt omgegaan;
- Welke waarborgen heeft de beheerafdeling dat eventuele storingen verholpen kunnen worden;
- Blijven de databases wel binnen grenzen die voor de beheerafdeling hanteerbaar zijn;
- Hoe gedraagt het systeem zich als het in gebruik genomen wordt;
- Wordt er bijvoorbeeld niet een te groot beslag beslag gelegd op systeemresources?

De PAT is een test die door de beheerafdeling wordt gedaan. Veelal heeft de beheerafdeling al wel formele of informele afspraken gemaakt die aangeven wanneer men tot beheer kan overgaan. Soms kan de beheerafdeling vanaf de start van het datawarehouse-traject betrokken worden. Dit heeft als voordeel dat tijdens de ontwikkeling geen initiatieven ontstaan die tegen wensen van de beheerafdeling ingaan. Daarmee wordt acceptatie door beheer een stuk eenvoudiger dan in een situatie waarbij een stuk software 'over de muur wordt gegooid'.

Conclusies

Een strategische aanpak van testen is noodzakelijk om aan de specifieke eisen die een datawarehouse stelt tegemoet te komen en om de daarmee samenhangende risico's in voldoende mate af te dekken. De strategie kent verschillende testfases, die zich ieder richten op een duidelijk gespecificeerd deel of aspect van het op te leveren systeem. Daarbij zal elke testfase eigen bevindingen produceren; elke test heeft een eigen doelgebied.
Bij de methode die geënt is op het V-model van testen wordt men steeds geconfronteerd met een deelprobleem, waardoor het geheel overzichtelijk blijft. Er ontstaan aldus testgebieden die goed te overzien zijn, met een minimale overlap en wordt dubbel werk voorkomen. Uiteindelijk is op deze wijze het gehele datawarehouse te testen.

Tom van Maanen (tom.van.maanen@capgemini.com) en **Eli Diemer** zijn werkzaam bij Capgemini.