Samenwerken en automatiseren versnelt testfase

Datawarehouse testen: van theorie naar praktijk

Marianne Kompagne en Chun Bon Tse

Voor het ontwerpen van een datawarehouse geldt vaak het advies "denk groot, maar begin klein". Voor het testproces van een datawarehouse geldt eigenlijk hetzelfde. Een testproces moet immers mee kunnen groeien met de groei van een datawarehouse.

Tijdens deze groei is het belangrijk om het testproces voortdurend te evalueren en te verbeteren. De auteurs van dit artikel voerden een onderzoek uit naar de mogelijkheden om het testproces van een datawarehouse te verbeteren. Dit artikel is een verslag van de zoektocht en bevindingen, met daarin vooral aandacht voor een verbetering van de testbasis door samenwerking van belanghebbenden, een transparant testproces, een kortere testdoorlooptijd en een kwalitatief hoge testoutput.

Het testproces krijgt tijdens de groei van het datawarehouse te maken met meer testers, een beheerafdeling inrichting en incrementele opleveringen van het datawarehouse. Dit zijn allemaal zaken die het testproces zodanig kunnen beïnvloeden dat er vertraging of verstoring optreedt.

Bij het opzetten van een testproces wordt veelal gebruik gemaakt van een tweetal erkende standaarden. Voor het testen maakt men vaak gebruik van TMap (zie afbeelding 1). Het project management wordt meestal ingericht volgens de richtlijnen van Prince II. Deze standaard richt zich specifiek op de procesmatige projectaanpak met aanpassingen naar aanleiding van ervaringen. Vervolgens is het raadzaam om gebruik te maken van het CMMI level 2 volwassenheidsmodel om het proces transparant te houden. De kracht van dit model zit in transparantie en requirements tracking.

Evaluatie

Uit de evaluatie van het testproces van een datawarehouse kwam een aantal verbeterpunten naar voren. Deze verbeterpunten zitten duidelijk niet in de kwaliteit of de nauwkeurigheid van het opgeleverde datawarehouse, maar hangen nauw samen met de gevolgen van de groei van een datawarehouse en de hoeveelheid extra tijd die daarbij komt kijken.

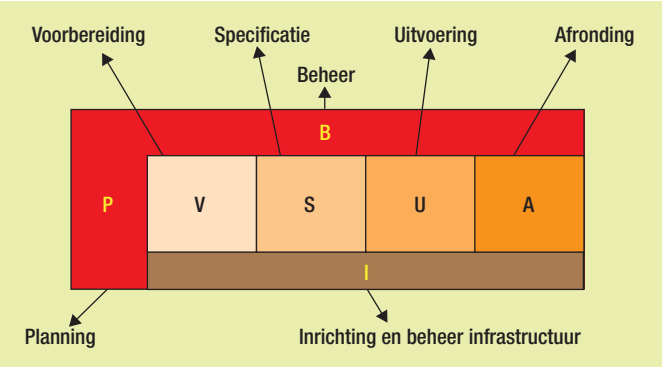
Complexe controles. Het eerste knelpunt waarmee de onderzoekers geconfronteerd werden was dat de voorbereiding van een systeemtest van een datawarehouse increment te veel tijd in beslag neemt. Het samenstellen van controle query's in het bijzonder is erg tijdrovend. De complexiteit van de controles neemt bovendien toe naarmate er aan de geïntegreerde datawarehouse-omgeving voortdurend incrementen worden aangebouwd.

Toename doorlooptijd regressietest. Tweede struikelblok was dat door de groei van het aantal incrementen de doorlooptijd van de regressietest toenam. De combinatie van incrementele opleveringen aan een geïntegreerde datawarehouse area-laag en een datamarts-laag zorgt ervoor dat bestaande ETL-processen bij een nieuw increment gewijzigd kunnen zijn, waardoor er weer opnieuw een regressietest uitgevoerd moet worden.

Gebrek samenhang tussen oplevering en testscripts. Derde punt is dat er geen goede correlatie is tussen opleveringen en testscripts. Hierdoor is het voor anderen dan de tester niet inzichtelijk welke versie van een testscript een bepaald testresultaat heeft opgeleverd en op welke oplevering dit testresultaat betrekking heeft. Bovendien zijn de testscripts en testresultaten niet centraal toegankelijk. Hierdoor is hergebruik van de testscripts door andere testers erg lastig.

Te weinig kruisbestuiving. Het gebrek aan voldoende testkennis bij ontwikkelaars en voldoende ontwikkelkennis bij de testers is een vierde knelpunt. Hierdoor wordt kennis van ontwikkelaars te weinig gebruikt bij het maken van testscripts en worden testscripts van opgeleverde incrementen niet gebruikt tijdens de ontwikkelfase voor controledoeleinden.

Onvolledige testbasis. In de evaluatie is ook de testbasis nader bekeken. In deze testbasis ontbreekt het eigenlijk veelal aan technische ontwerpen. De kwaliteit van de onderdelen die wel aanwezig zijn, zoals requirements, functioneel ontwerpen en bouwconventies, is erg hoog. De borging van de requirements en de functioneel ontwerpen binnen de test kunnen echter wel verbeterd worden.



Afbeelding 1: De TMap fasen in een testproces.

Onvolledige testsets. Het zesde en laatste struikelblok is tijd. Door de werkdruk is het moeilijk om de bij het project behorende testset actueel te houden. Verder is gekeken waar tijdwinst te behalen is. Zo is geconstateerd dat in het project bevindingen in de opgeleverde datastructuren meer werk en vertragingen in doorlooptijd tot gevolg hebben dan bevindingen in de ETL-processen. Denk bijvoorbeeld aan het ontbreken van verplicht stellen van een veld in een datastructuur. Deze zijn snel te herstellen, maar hebben een grote impact op de doorlooptijd van het testproces, doordat hierbij de testset in de testomgeving vaak opnieuw opgebouwd moet worden.

De evaluatie biedt voldoende informatie om tot een overzicht te komen met de belangrijkste voorwaarden voor het ideale testproces, namelijk:

- Kwaliteitoptimalisatie van zowel de ontwikkelingsfase als de testfase;
- Reduceren van het aantal bevindingen door de kwaliteit van de testbasis te verhogen (requirements en ontwerpen);
- Meer aandacht besteden aan unittest en daarin strenger controleren op bouwconventies;
- Verkorten van de testdoorlooptijd voor een opgeleverd datawarehouse increment (inclusief regressietest) zonder de eerder genoemde kwaliteit geweld aan te doen;
- Verhogen van de transparantie van het testproces door de uitgevoerde testen en testresultaten eenduidig vast te leggen;
- Laagdrempelige overdracht naar de beheerorganisatie;
- Zoveel mogelijk automatiseren van de testuitvoering en testresultaatopslag.

Om tot het ideale testproces voor datawarehouses te komen stellen we een aantal punten ter verbetering van het testproces voor. Deze voorstellen hebben betrekking op de testbasis en unittests, en geven een voorzet voor de verbetering van de systeemtest (testapplicatie) en Functionele Acceptatie Test (reviews). De acties zullen volgens ons leiden tot een algehele kwaliteitsverbetering van het proces. Een plan van aanpak volgt.

Verbeteren testbasis

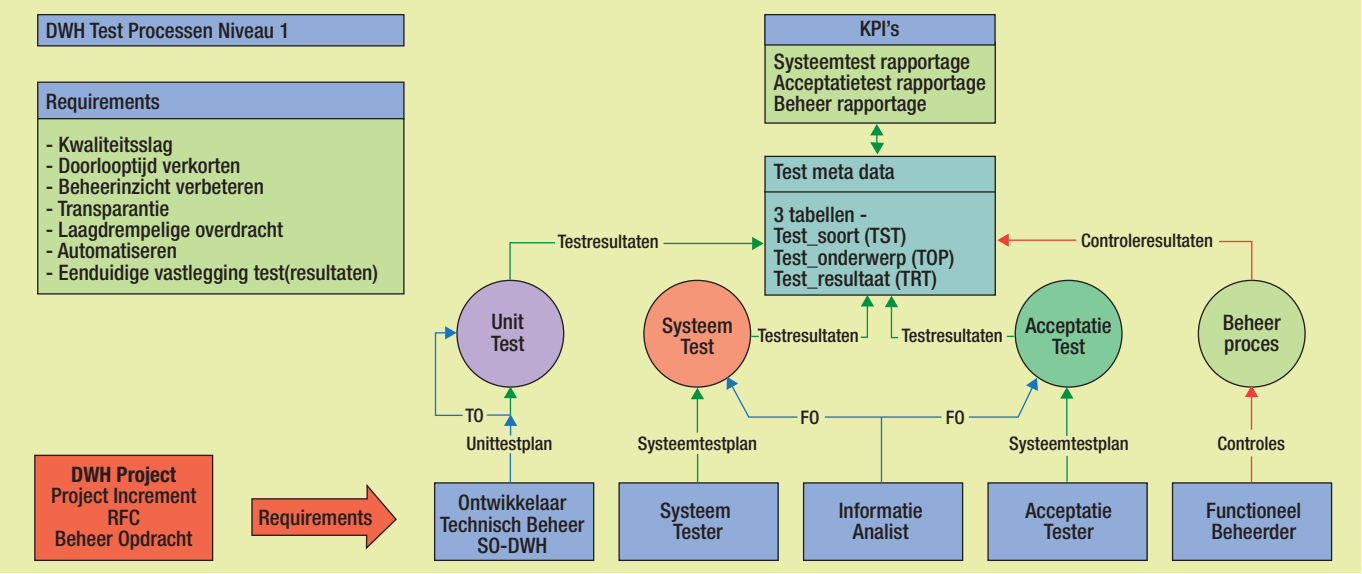
Om tot verhoging van de kwaliteit van de requirements en de functioneel ontwerpen te komen, dient een aantal verbeteringen te worden doorgevoerd. Naast brainstormsessies waarin ontwerpers een overall functioneel ontwerp in grote lijnen opstellen, zijn er plenaire reviews van functioneel ontwerpen per datawarehouse increment met ontwikkelaars, testers, beheer en de business nodig om het ontwerp op alle aspecten te beoordelen. Hierbij wordt gebruik gemaakt van use cases.
De functioneel ontwerpdocumentatie dient zo opgezet te zijn dat deze een ingang heeft per datawarehouse increment voor projectdoeleinden, plus een ingang voor het overall datawarehouse functioneel ontwerp voor beheerdoeleinden. Ten behoeve van transparantie kunnen er verwijzingen (CMMI compliant) in de functioneel ontwerpen worden opgenomen naar op te lossen requirements. Daarnaast moeten er ten behoeve van de ondersteuning bij testdefinities beschrijvingen van functionele controletellingen in de functioneel ontwerpen worden opgenomen. Door gebruik te maken van templates ontstaat er een eenduidige manier om de functionele ontwerpen ook daadwerkelijk vast te leggen. Door deze maatregelen zal de kwaliteit van de testdefinities en de testscripts omhoog gaan en neemt de doorlooptijd van het maken van testdefinities af. De verbeteringen zijn daarnaast leuk om uit te voeren, zodat ze de betrokkenheid en samenwerking van alle deelnemers verhogen. En passant zorgen ze ook nog voor een beter acceptatie.

Unittest

Met in het achterhoofd de filosofie 'Hoe hoger de kwaliteit van de input in het testproces des te hoger is de kwaliteit van de output uit het testproces' hebben de onderzoekers ook gekeken naar de verbetering van de unittest. Bij deze test is met speciale aandacht gekeken naar de correctheid en compleetheid ten aanzien van datastructuren en bouwconventies. Door zo compleet en correct mogelijk te werken worden fouten in de ontwikkelingsfase eerder geconstateerd en opgepakt, wat leidt tot een vermindering van het aantal bevindingen in de systeemtest.
Voorbeelden van activiteiten waarmee veel tijdwinst te behalen is, zijn: het zorgen voor correcte naamgeving van objecten en de

Lagen in datawarehouse

Een datawarehouse bestaat uit een aantal datamarts die incrementeel ontworpen en opgeleverd worden. De datawarehouse architectuur bestaat uit drie lagen: een staging area, een datawarehouse area met historie tijdstroken en een datamarts area (ook wel informatie area genoemd). Per laag wordt de invulling van de verschillende functionaliteiten beschreven. In de stage area laag kunnen gegevens uit interne en externe bronnen worden ontvangen. De datawarehouse area is bedoeld om gegevens voor lagere tijd vast te houden. Het kan gezien worden als het geheugen voor lange termijn . De informatie area laag is bedoeld om kennisintensieve processen van de organisatie te ondersteunen. De representatie van data in deze laag is begrijpelijk voor eindgebruikers.



Afbeelding 2: Grafische weergave van het ideale testproces.

correcte implementatie van data-objecten. Goed uitvoeren van de unittest vereist discipline van de betrokkenen, ook onder grote tijdsdruk. De tijd die besteed wordt aan unittesten wordt dubbel en dwars terugverdiend op de doorlooptijd van het systeemtesten. Een neveneffect is dat er meer aandacht is voor functionele testresultaten.

Verbeteringen systeemtest

Het maken van een testapplicatie levert niet alleen voordelen voor het proces, het voegt ook een leuk element toe aan het proces als geheel. Als proef hebben de onderzoekers een klein ontwerp gemaakt, waarna een ontwikkelaar de applicatie heeft gebouwd. Hierbij hebben de onderzoekers rekening gehouden met hergebruik van de al aanwezige SQL testscripts. Aangezien deze testscripts een verschillend aantal resultaatkolommen hadden, is gekozen voor een applicatie die een SQL statement afvuurt en het resultaat in verschillende resultaattabellen opslaat. Deze resultaattabellen hebben met elkaar minstens één kolom gemeen. Ze moeten allemaal een 'resultaat' kolom met een waarde 'OK' of 'NOK' hebben. Dit is nodig om een overall testrapportage te kunnen maken.

De testapplicatie bestaat verder uit een 'onderwerp' tabel met de definitie van de uit te voeren test. Er is een 'resultaat' tabel met informatie over de uitgevoerde test inclusief SQL scripts en er zijn per test resultatentabellen die dynamisch bij de eerste uitvoering van een test aangemaakt worden. Er is een package die het afvuren van een SQL testscript mogelijk maakt en voor het opslaan van de resultaten zorgt. Een view op alle 'resultaat'-kolommen zorgt er voor dat er inzicht is in het resultaat van een batchverwerking. Bij de bevindingen geven de detailtabellen van het testresultaat voldoende informatie om analyses mogelijk te maken. Door het testproces te ondersteunen met een testapplicatie wordt er allereerst voor gezorgd dat testresultaten opgeslagen worden in de testapplicatie. Dit zorgt voor beter zichtbare testresultaten (transparantie) en de hieruit te trekken lessen zullen duidelijker naar voren komen. Door gebruik te maken van herbruikbare rapporten op de test-metadata kunnen testrapportages bovendien sneller opgeleverd worden. Ten tweede maakt de inzet van een testapplicatie hergebruik van testscripts mogelijk. Door de opgestelde testen of controles op te slaan in de testapplicatie kunnen deze bij volgende opleveringen van datawarehouse incrementen hergebruikt worden als onderdeel van een regressietest. Dit zorgt voor een betere garantie van de betrouwbaarheid en de kwaliteit. Verder zal, met opgeslagen testresultaten op zowel hoog als laag detailniveau, het voor zowel de testers als voor datawarehousebeheer duidelijk zijn hoe de datawarehouseprocessen zich gedragen. De resultaten van de controletellingen geven dan inzicht in de groei van datamarts en de aantallen van de aangeleverde brondata kunnen gevolgd worden.

Automatisering

De doorlooptijd van de systeemtest neemt van het hele testproces de meeste tijd in beslag. Met een testapplicatie wordt het mogelijk gemaakt om een reeks testen automatisch uit te voeren. Zo kan bijvoorbeeld een stap als het testen van de datastructuren, zoals de data-objectstructuur, de domeinen, foreign key relaties, automatisch worden uitgevoerd. Ook het testen van functionaliteit van de ETL-processen is met het bouwen van een testapplicatie geautomatiseerd.

Testers dienen er wel rekening mee te houden dat er een inwerkperiode aan de systeemtest vastzit die de resultaten in het begin zal beïnvloeden. Bij goed ingewerkte testers kan de doorlooptijd verder verbeteren. Schematisch is het model van het besproken testproces weergegeven in afbeelding 2.

Marianne Kompagne CBIP is DWH Architect bij Kadenza.
Chun Bon Tse is DWH Ontwikkelaar bij Kadenza.