TESTEN IN ONDERHOUD

auteur: Leo van der Aalst
gebaseerd op de originele white paper

TESTEN IN ONDERHOUD

auteur: Leo van der Aalst
gebaseerd op de originele white paper

1 INLEIDING

Vanaf de jaren tachtig groeit de ontwikkeling en het gebruik van informatiesystemen en technische infrastructuren meer dan ooit. De afhankelijkheden tussen de bedrijfsprocessen enerzijds en de geautomatiseerde informatievoorziening anderzijds worden alsmaar sterker. Ook moeten bestaande informatiesystemen steeds vaker aangepast worden aan nieuwe wensen. Sinds het begin van de jaren negentig groeit het besef, dat de kosten tijdens gebruik en beheer van informatiesystemen ver uitstijgen boven de initiële ontwikkelkosten.

De bestaande literatuur over testen en testmethoden geeft voldoende houvast voor het testen van nieuwbouw software applicaties. In de praktijk blijkt dat het niet altijd voor iedereen even eenvoudig is om een dergelijke aanpak te vertalen naar het testen van software in een onderhoudssituatie.

Als we uitgaan van een gemiddelde "levensduur" van tien jaar [Tamai 1992] voor een software applicatie dan is veilig aan te nemen dat er vaker in een onderhoudssituatie dan in een nieuwbouwsituatie wordt getest. Hierbij is het dan ook niet verrassend als Grady laat zien dat er meer geld is gemoeid met onderhoud dan met nieuwbouw van een software applicatie [Grady 1987]. Het is daarom des te opvallender dat er in de literatuur weinig te vinden is over dit onderwerp. Met dit document denken we dan ook in de behoeften te voorzien van velen die regelmatig met het testen in een onderhoudssituatie te maken hebben. Hierbij hanteren we de gestructureerde testmethodiek TMap als basis en geven we de verschillen aan voor onderhoudstesten. Bij de lezer wordt kennis van TMap dan ook bekend verondersteld.

Testen in een onderhoudssituatie is uiteraard geen nieuw werkterrein. Sterker nog; het is voor velen hun dagelijks werk. Dit heeft als voordeel dat er veel praktijkervaringen zijn op dit gebied en deze zijn dan ook in dit document verwerkt. Enkele aansprekende voorbeelden hiervan zijn:

- Organiseren van een *kick-off sessie* met alle relevante partijen voor het verkrijgen van helderheid over uiteenlopende onderwerpen.
- Maken van een *standaard onderhoudstestplan* met daarin alle herbruikbare testaspecten.
- Het opzetten, gebruiken en onderhouden van een *schaalbare regressietestset*.

De volgende mensen worden hartelijk bedankt voor hun medewerking en bijdragen tot het totstandkomen van dit document: Rob Baarda, Eric Begeer, Martin Boomgaardt, Guy Holtus, Tim Koomen, Peter van Lint, Marc Valkier en Michiel Vroon.

2 BEHEER EN ONDERHOUD

Beheer en onderhoud worden vaak in één adem genoemd zonder stil te staan bij wat iemand daar onder verstaat. Voordat kan worden ingegaan op testen in onderhoudssituaties wordt in dit hoofdstuk toegelicht wat wordt verstaan onder beheer en onderhoud. Daarnaast wordt een mogelijke relatie tussen de diverse beheervormen en een organisatorische indeling gegeven en wordt kort stilgestaan bij de levensduur van een informatiesysteem.

2.1 Wat is beheer

Het beheer is in veel organisaties volgens het model van Mintzberg [Delen 1992] op zowel strategisch-, tactisch- als operationeel niveau ingevuld.

Op strategisch niveau worden beslissingen genomen die van strategisch belang zijn voor het beheer, bijvoorbeeld "het beheer wordt zoveel mogelijk uitbesteed buiten het bedrijf". Op tactisch niveau bevinden zich de taken die gerelateerd zijn aan het beschikbaar stellen van materiële, personele en financiële middelen met betrekking tot het beheer. Op operationeel niveau tenslotte bevinden zich de taken die essentieel zijn voor de directe uitvoering van het beheer en daarom wordt dit niveau nader toegelicht. Het totale beheer van de informatievoorziening is op operationeel niveau op te splitsen in drie beheervormen:

- Functioneel beheer
- Technisch beheer
- Applicatiebeheer

Functioneel beheer

Functioneel beheer omvat alle beheertaken die nodig zijn voor het dagelijks gebruik van informatiesystemen, de gegevensinfrastructuur en wijziging van de specificaties daarvan.

In de dagelijkse praktijk omvat functioneel beheer het volgende:

- Het vormen van de schakel tussen gebruikers en beheer van de informatievoorziening.
- Het begeleiden en opleiden van gebruikers met betrekking tot het gebruik van informatiesystemen.
- Het toekennen van autorisaties.
- Het beheer van applicatiegebonden gegevens.
- Het inhoudelijk beheer van gegevensverzamelingen.
- Het bewaken van het juiste gebruik van het informatiesysteem.
- Onderhouden van de handmatige procedures.
- Onderhouden van de functionele specificaties.
- Plannen en uitvoeren van een (gebruikers)acceptatietest.

Technisch beheer

Technisch beheer omvat alle beheertaken die nodig zijn voor het installeren en operationeel maken en houden van informatiesystemen en technische infrastructuren.

Tot het technisch beheer horen onder andere de volgende zaken:

- Beschikbaar stellen en onderhouden van pakketten (o.a. tekstverwerking, electronische agenda) voor persoonlijke ondersteuning.
- Beschikbaar stellen en onderhouden van informatiesystemen voor de ondersteuning van groepswerk.
- Een helpdesk functie.
- Onderhouden van de technische infrastructuur.
- Aanbieden van verwerkingscapaciteit.
- Aanbieden van opslagcapaciteit.
- Plannen en uitvoeren van een (productie)acceptatietest.
- Plannen en uitvoeren van unit-, integratie- en systeemtests op *technische* infrastructuur.

Het technisch beheer is dus vooral gericht op het technisch platform, bestaande uit apparatuur met bijbehorende basisprogrammatuur, en de operationalisering van de hierop gebouwde informatiesystemen.

Applicatiebeheer

Tot applicatiebeheer horen onder andere de volgende zaken:

- Beheer applicatiebibliotheek.
- Beheer gegevensbanken.
- Wijzigen applicatieprogramma's.
- Wijzigen gegevensbanken.
- Plannen en uitvoeren van unit-, integratie- en systeemtest op applicaties.

Ook als er geen directe aanleiding is om applicatie-onderhoud uit te voeren, is het vanwege de continuïteit toch noodzakelijk om een volledige applicatiebeheervorm in stand te houden, inclusief management en staftaken.

2.2 Wat is onderhoud

Een onderdeel van applicatiebeheer dat op operationeel niveau voorkomt noemen we onderhoud. Onderhoud is het wijzigen of aanvullen van een bestaand informatiesysteem en kan *ad-hoc* of *planmatig* worden uitgevoerd.
Ad-hoc onderhoud vindt plaats bij het oplossen van fouten die geen uitstel kunnen verdragen, omdat deze onacceptabele schade aanrichten in productie. Hieronder valt alleen correctief onderhoud.
Onder planbaar onderhoud vallen alle overige soorten onderhoud. Deze worden conform reguliere ontwikkelprocessen releasematig uitgevoerd, waarbij meestal wordt gestart met een impactanalyse. Er zijn ook vormen van correctief onderhoud die planmatig uitgevoerd kunnen worden. Dit betreft fouten, die niet direct opgelost hoeven te worden, omdat ze acceptabele- of geen schade aanrichten in productie.

Onderstaand figuur laat zien welke vijf soorten onderhoud [Delen 1992] onderscheiden worden.

	Ad-hoc	Planmatig
In stand houden	correctief	correctief
		preventief
		adaptief
Verbeteren		perfectief
Aanvullen		functionaliteit

komt niet voor meest voorkomend

Correctief onderhoud is het herstellen van afwijkingen in componenten van het informatiesysteem. Afhankelijk van het object van onderhoud kan dit variëren van het verwijderen van bugs uit de applicatieprogrammatuur tot het verhelpen van storingen in de apparatuur. Bij correctief onderhoud blijven het beoogde gebruik en de exploitatie van de informatievoorziening ongewijzigd.

Preventief onderhoud is het corrigeren van componenten van het informatiesysteem, zonder aanleiding in de vorm van een probleemrapport. Doel van preventief onderhoud

is: (a) het voorkomen van toekomstige problemen, of (b) het verhogen van de onderhoudbaarheid.

Adaptief onderhoud is het aanpassen van één of meer delen van het informatiesysteem als gevolg van wijzigingen in de omgeving van die delen. Adaptief onderhoud kan worden veroorzaakt door onderhoud aan een ander deel (meestal in een onderliggende laag) van het informatiesysteem, door wijziging van een ander informatiesysteem waarmee een interface bestaat, of door gewijzigde regelgeving voor de bedrijfsfunctie die wordt ondersteund.

Perfectief onderhoud is het aanpassen van een component van het informatiesysteem aan veranderde kwaliteitseisen van de gebruikers.

Functioneel onderhoud omvat het uitbreiden of wijzigen van de functionaliteit van het informatiesysteem.

2.3 Beheerorganisaties

In deze paragraaf worden de drie beheervormen vertaald naar drie soorten organieke beheereenheden. Er worden vaak drie soorten organisaties onderscheiden, te weten:
- de gebruikersorganisatie;
- de verwerkingsorganisatie;
- de systeemontwikkel- en onderhoudsorganisatie.

Hier worden de beheervormen als volgt aan toegekend:
Functioneel beheer -> gebruikersorganisatie
Technisch beheer -> verwerkingsorganisatie
Applicatiebeheer -> onderhoudsorganisatie

De gebruikersorganisatie is het geheel van mensen en middelen dat zich bezighoudt met de bedrijfsprocessen. Binnen een dergelijke gebruikersorganisatie treft men dan ook in de eerste plaats primaire bedrijfsfuncties aan en pas in de tweede plaats ondersteunende zaken als de gebruiks- en beheerfuncties ten aanzien van de informatievoorziening. Met beheerfuncties wordt het functioneel beheer bedoeld, dat vanwege de materiekennis, de verantwoordelijkheid en de kennis van de eigen gebruikersorganisatie die ermee gemoeid zijn, niet kan worden uitbesteed. Meestal gaat men uit van een aparte beheerorganisatie binnen de gebruikersorganisatie, omdat zoals eerder gezegd, de gebruikersorganisatie zich niet alleen met beheer bezig houdt.

Een verwerkingsorganisatie is het geheel van mensen en middelen dat zich bezighoudt met het operationeel maken en houden van informatiesystemen en de technische infrastructuur. Dit soort organieke eenheden kan variëren van een groot reken-, communicatie- en servicecentrum tot een kleinschalige verwerkingsorganisatie op een afdeling binnen de gebruikersorganisatie. Gezien de goed te omschrijven werkzaamheden van een verwerkingsorganisatie besluiten veel organisaties ertoe om dit werk uit te besteden.

Een onderhoudsorganisatie is het geheel van mensen en middelen dat zich bezighoudt met het ontwikkelen en onderhouden van applicaties. Binnen een onderhoudsorganisatie komen meestal aparte eenheden voor ten behoeve van applicatie-ontwikkeling en applicatie-onderhoud. Een verschil tussen beide is, dat ontwikkeling meestal wordt uitgevoerd door een eenmalige projectorganisatie, terwijl de organisatie voor het onderhoud permanent is. Net als eerder genoemd bij de verwerkingsorganisatie gebeurt het ook steeds vaker dat de werkzaamheden van een onderhoudsorganisatie worden uitbesteed.

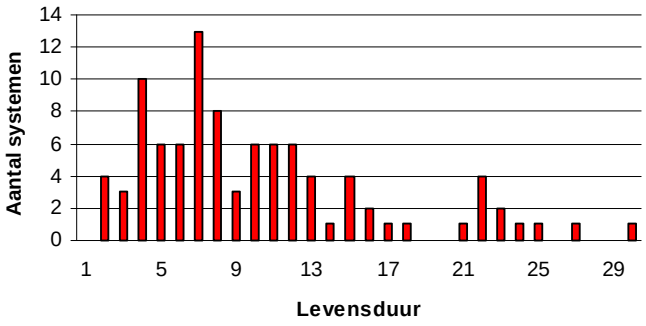
Samenwerking tussen de beheereenheden

Het functioneel beheer, het applicatie-onderhoud en het technisch beheer kunnen binnen een en hetzelfde bedrijf op diverse plaatsen afspelen en bepaald beheer kan zelfs zijn uitbesteed. Het is daarom van groot belang dat de drie beheerorganisaties onderling samenwerken. Bij het wijzigen van een informatiesysteem is nauwe samenwerking vereist tussen deze organisaties en eventueel de partij waaraan werk is uitbesteed.
Er wordt onderscheid gemaakt tussen de volgende twee categorieën van wijzigingen:

- Wijzigingsbeheer
  Hieronder valt adaptief-, perfectief- en functioneel onderhoud. Bij deze onderhoudssoorten verandert het gebruik en/of de exploitatie van de informatievoorziening en is gezamenlijke besluitvorming door het management van alle betrokken partijen vereist.
- Probleembeheer
  Hieronder valt correctief- en preventief onderhoud. Hierbij veranderen het gebruik en de exploitatie van de informatievoorziening niet. Daarom is coördinatie op operationeel niveau, zoals een helpdesk, hier veelal voldoende.

2.4 Levensduur van een informatiesysteem

Onderhoud kost geld. Het is voor een onderneming daarom belangrijk om te bepalen wanneer onderhoud nog lonend is. Na verloop van tijd is het namelijk vaak goedkoper om een nieuw informatiesysteem te laten bouwen in plaats van het informatiesysteem te blijven onderhouden. Vlak na de ingebruikneming is vaak een verhoogd onderhoud te zien door allerlei kinderziektes die nog in het informatiesysteem zitten (correctief onderhoud). Gedurende de levensloop van het informatiesysteem nemen de echte fouten hopelijk af maar beginnen de andere vier vormen van onderhoud toe te nemen. Door onderhoudswerkzaamheden zal in het algemeen de structuur van de programmatuur langzamerhand steeds slechter worden, waardoor de onderhoudbaarheid zal afnemen. Voorts is onderhoud net als nieuwbouw nog steeds mensenwerk dus is er een kans dat er nieuwe fouten worden geïntroduceerd. De verslechterende structuur en de nieuw geïntroduceerde fouten geven zogenaamde ouderdomsverschijnselen waardoor op een gegeven moment wordt beslist dat het onderhavige informatiesysteem economisch gesproken beter aan zijn einde kan komen. Uit onderzoek van Tamai met betrekking tot 95 systemen blijkt dat de gemiddelde levensduur van een informatiesysteem 10 jaar bedraagt [Tamai 1992]. Hierbij zijn variaties waargenomen van 2 tot 30 jaar (zie figuur 1, levensduur informatiesystemen).



Figuur 1, levensduur informatiesystemen.

3 ENKELE TESTAANDACHTSPUNTEN BIJ ONDERHOUD

Na het hoofdstuk met de toelichting op beheer en onderhoud en voor het hoofdstuk wat het testen in onderhoud beschrijft, wordt hier nader ingegaan op specifieke testaandachtspunten bij onderhoud.

- Planbaar en niet-planbaar onderhoud.
- Ontwikkelproces voor onderhoud.
- Uitdagingen voor de beheerorganisaties.

In paragraaf 3.4 worden verwijzingen naar oplossingen gegeven.

3.1 Planbaar en niet-planbaar onderhoud

Gepland

90% van het totale onderhoud op een informatiesysteem is gepland. En omdat een informatiesysteem zich langer, denk aan de eerder genoemde levensduur, in een onderhoudssituatie bevindt dan in een nieuwbouwsituatie en dus ook langer in een onderhoudssituatie wordt getest, is het van belang hier een gestructureerde testaanpak voor te kiezen.

Niet-gepland

Bij de resterende 10% gaat het om niet-gepland (ad-hoc), vooral correctief onderhoud. In dit geval is het duidelijk dat het oplossen van een productieprobleem alle prioriteit heeft. Voor het uitvoeren van een risicoanalyse, het opstellen van een teststrategie en zelfs voor het maken van testgevallen is in het algemeen weinig of zelfs geen tijd beschikbaar. Toch wil men zo beheerst mogelijk in productie met de gewijzigde software. Voor de tester ligt hier een uitdaging om ook bij, niet planbaar, ad-hoc onderhoud een kwaliteitsoordeel te kunnen geven over de doorgevoerde wijziging.
Niet altijd hoeft correctief onderhoud direct plaats te vinden. Soms bestaat het oplossen van een productieprobleem uit het verhelpen van de gevolgen. Bijvoorbeeld uit het corrigeren van data of zorgen dat gebruikers weer verder kunnen. Het structureel corrigeren van het probleem vindt dan later plaats en is planbaar.

3.2 Ontwikkelproces voor onderhoud

Het ontwikkelproces voor onderhoud verschilt, hoewel niet veel, van die in nieuwbouwsituaties. De verschillen bevinden zich vooral in de initiatie van het proces en in het aandeel testen. Onderhoud wordt vaak geïnitieerd door een wijzigingsaanvraag (denk aan wijziging in; functionaliteit, middleware, pakket) of een releaseplan en tevens is het relatieve testaandeel bij onderhoud vaak groter dan bij nieuwbouw. Ook al verschilt het proces niet veel met die in nieuwbouwsituaties, toch zal deze invloed kunnen hebben op de te volgen teststrategie. Immers bij nieuwbouw wordt het systeem nieuw gebouwd en wordt "alles" getest. Bij onderhoud vinden echter wijzigingen plaats op een bestaand systeem. Vragen hierbij zijn of het dan voldoende is om alleen de wijzigingen te testen en hoe omgegaan moet worden met eventueel al aanwezige testware.
Extra lastig wordt het als een onbeheersbare situatie dreigt te ontstaan doordat de opleveringen van het onderhoudsteam uit deeloplossingen blijken te bestaan, die ook nog eens verspreid in de tijd worden opgeleverd.
Hiernaast komt ook de situatie voor dat er tijdens een nieuw- c.q. aanbouwtraject geen rekening wordt gehouden met parallel daaraan in onderhoud doorgevoerde wijzigingen op het productiesysteem. Dit kan onder andere leiden tot gaten en/of overlap in testwerkzaamheden.

3.3 Uitdagingen voor de beheerorganisaties

In tegenstelling tot nieuwbouw, waarbij sprake is van een éénmalige projectorganisatie, wordt onderhoud meestal belegd in een onderhoudsorganisatie (applicatiebeheer). Een complicerende factor hierbij is dat naast deze beheereenheid ook nog de functioneel- en technisch beheerorganisaties bestaan. Immers functioneel beheer "levert" de functionele specificaties aan onder andere applicatiebeheer en is vaak verantwoordelijk voor het uitvoeren van een (gebruikers)acceptatietest. Applicatiebeheer is op haar beurt verantwoordelijk voor het doorvoeren van de aanpassingen in de software en het uitvoeren van een programma-/systeemtest. Daarnaast is technisch beheer bijvoorbeeld verantwoordelijk voor de testinfrastructuur en het uitvoeren van een productie acceptatietest. Het is van groot belang dat deze organisaties op een duidelijke manier samenwerken. Belangrijk hulpmiddel hierbij is het uitvoeren van een impactanalyse op elke wijzigingsaanvraag, waaruit duidelijk moet worden wat (infrastructuur, specificaties, applicaties, testware, enz.) aangepast moet worden en wie wat moet doen
Hiernaast moeten ze gezamenlijk een aantal (test)uitdagingen oplossen op de gebieden:

Keuzes in testinfrastructuur

Op het gebied van testinfrastructuur moeten keuzes worden gemaakt in het wel of niet kunnen beschikken over een permanente testomgeving met daarin een testdatabase met wel of niet actuele productiedata. Ook moet er worden nagedacht over de testomgeving die noodzakelijk is voor met name, niet-gepland, ad-hoc onderhoud.
Verder moet worden onderzocht of geautomatiseerd testen een optie is. Een uitdaging hierbij is het kunnen beschikken over een goede testdatabase. Een mogelijke conflictsituatie kan hierbij zijn dat voor het testen van een wijziging een testdatabase nodig is met actuele (productie)data, terwijl voor geautomatiseerd testen juist een goed gedefinieerde en stabiele testdatabase gewenst is. Om een conflictsituatie te voorkomen moet goed worden nagedacht over de te gebruiken testdatabase en hoe deze gevuld en/of actueel gehouden moet worden.

Testwarebeheer

Voor het beheren en onderhouden van testware wordt niet altijd tijd vrijgemaakt. Bij ad-hoc onderhoud komt daar nog bij dat het oplossen van een productieprobleem, overigens terecht, prevaleert boven het bijwerken van de testware. In het algemeen moeten keuzes worden gemaakt of, en zo ja hoe, de testware actueel gehouden kan worden. Daarnaast is het gewenst om over een regressietestset te kunnen beschikken. Net als bij de testware in het algemeen moet ook hier goed worden nagedacht over wanneer, hoe en door wie deze testset actueel gehouden kan worden.

Materiekennis noodzakelijk

Het is niet eenvoudig om een testgeval te maken dat precies het opgeloste productieprobleem test of die een doorgevoerde wijziging test. Hier is niet zelden diepgaande materiekennis van het systeem voor nodig ook al omdat systeemdocumentatie vaak niet aanwezig of verouderd is. Het vraagt veel kennis om precies die uitgangsituatie in de testdatabase te creëren, die noodzakelijk is om het testgeval te kunnen uitvoeren. Materiekennis is niet altijd voldoende aanwezig in beheerorganisaties. Vaak is dit een groeipad, die meestal pas begint op het moment dat een systeem door een project aan de lijn wordt overgedragen. Indien materiedeskundigheid niet voldoende voorhanden is in de beheerorganisatie dan zal goed moeten worden nagedacht over hoe hier mee om te gaan bij het schrijven van testgevallen.

Demotivatie van testers

In nieuwbouwprojecten is vaak wel ruimte voor aparte testers. Tijdens onderhoud is deze ruimte er niet meer en moeten alle tests uitgevoerd worden door ontwikkelaars en beheerders. Deze hebben vaak een afkeer van testen vanwege het saaie, repetitieve

karakter. Dit kan leiden tot demotivatie van de beheerders om te gaan testen. Het is van belang om maatregelen te treffen om een dergelijke situatie te voorkomen.
Daarbij is het zo dat het oplossen van productieproblemen, terecht, voorrang heeft op het testwerk. Dit kan echter negatieve invloed hebben op de kwaliteit en de voortgang van de tests. Verder komt het regelmatig voor dat beheerders weinig tot geen affiniteit hebben met testen en elk excuus aangrijpen om met alles en nog wat bezig te zijn, als het maar niet met testen is.
Het is van belang deze verschijnselen te onderkennen en daar maatregelen voor te treffen.

Omvang van onderhoud

Onderhoud is er in alle soorten en maten. Variërend van kleine wijzigingen tot bij wijze van spreke een totaal redesign van het informatiesysteem. Bij omvangrijke wijzigingen zal de teststrategie anders zijn dan bij minder omvangrijke wijzigingen. Net als bij nieuwbouwtrajecten wordt ook hier regelmatig "vergeten" aandacht te geven aan niet-functionele kwaliteitsattributen.

Invloed van onderhoud op specificaties en testgevallen

Zoals in paragraaf 2.3 "beheerorganisaties" is beschreven wordt er onderscheid gemaakt tussen probleembeheer en wijzigingsbeheer. Bij probleembeheer veranderen de specificaties normaalgesproken niet. Dit in tegenstelling tot wijzigingsbeheer.
Complicerende factoren voor het testen zijn hierbij, dat bij probleembeheer voor de tester vaak onduidelijk is wat de concrete aanleiding was (oorzaak van het probleem) en hoe de fout gereproduceerd kan worden. Tevens is bij wijzigingsbeheer de wijziging niet altijd goed gedocumenteerd. Met name hoe en waar de wijziging is doorgevoerd wil nog wel eens onduidelijk zijn door het ontbreken van een impactanalyse.
In beide gevallen zal een bepaalde route gekozen moeten worden om te komen tot testgevallen. In de praktijk blijkt het niet eenvoudig te zijn om net die testgevallen te bedenken die de "vinger op de zere plek leggen". Dit blijkt vaak nogal specialistisch werk te zijn, die specifieke systeem- en/of materiekennis vereist.
Hiernaast geldt specifiek voor probleembeheer nog dat een bevinding niet alleen in de productieversie moet worden opgelost, maar ook moet worden verwerkt in alle daarop volgende softwareversies. Hier moet met het bepalen van de teststrategie rekening worden gehouden.

Oneigenlijke taken van testers

In een onderhoudssituatie komt het regelmatig voor dat testers met oneigenlijke, niet-specifieke, testtaken worden geconfronteerd. Dit komt vooral door het feit dat beheerders behalve testtaken ook beheertaken moeten uitvoeren. In sommige gevallen kan een combinatie van beheer- en testtaken geheel valide zijn. Maar het verdient aanbeveling dat testers zich bijvoorbeeld bij de volgende taken afvragen of deze wel als testen moeten worden gezien. Het probleem schuilt er namelijk niet zozeer in dat één persoon, bijvoorbeeld de functioneel beheerder, naast testen ook andere taken vervult. Waar het gevaar zit is dat de persoon taken moet vervullen waar deze niet geschikt voor is, of dat management een enorm aantal testuren ziet en vervolgens de verkeerde maatregelen gaat nemen om dit omlaag te krijgen. Voorbeelden van niet-specifieke testtaken:

- Testen in productieomgeving, omdat er geen testomgeving beschikbaar is.
- Bewaking van productietrends. Bijvoorbeeld het verzamelen van het aantal storingen of het aantal helpdesk calls per tijdseenheid.
- Analyse naar gevolgen/oorzaken van een storing, het informeren van gebruikers over doorgevoerde wijzigingen en de statusrapportage van het project aan de business zijn bijvoorbeeld taken die wel tot het takenpakket van een functioneel beheerder kan worden gerekend, maar wellicht niet als deze de rol van tester heeft.
- Bij het oplossen van een productieprobleem kan het zijn dat naast het aanpassen van de software en/of het schrijven van een correctieprogramma ook acties (denk aan

excuusbrief) richting medewerker/klant worden uitgezet. De vraag is of de tester met dit laatste iets te maken heeft.

3.4 Verwijzing naar oplossingen

In het volgende hoofdstuk wordt aangegeven hoe het testen in een onderhoudssituatie eruit moet zien. De hierboven geschetste aandachtspunten worden op meerdere plaatsen beantwoord.

Onderstaand is per aandachtspunt in een tabel aangegeven op welke plaatsen in het volgende hoofdstuk de bijbehorende oplossingsrichtingen zijn te vinden.

Testaandachtspunt		Oplossingsrichting(en)	
3.1 Gepland onderhoud		4.1	Specifiek proces Standaard onderhoudstestplan
		4.3	Permanente testomgeving
		4.4.1	Teststrategie
3.1 Niet-gepland onderhoud		4.1	Specifiek proces Standaard onderhoudstestplan
		4.3	Testomgeving op afroep beschikbaar
		4.4.1	Teststrategie
		4.4.4	Testkwaliteitverbetering bij ad-hoc onderhoud
3.2 Ontwikkelproces voor onderhoud		4.4.1	Vaste- en variabele testkosten Releasematig opleveren Teststrategie
3.3 Uitdagingen voor de beheerorganisaties		4.2	Planning, prio, structuur
	Keuzes in testinfrastructuur	4.3	Permanente testomgeving Testomgeving op afroep beschikbaar Testdatabase
	Testwarebeheer	4.4.3	Schaalbare regressietestset
		4.4.4	Testkwaliteitverbetering bij ad-hoc onderhoud
		4.4.5	Regressietestset aanpassen
	Materiekennis noodzakelijk	4.4.1	Kick-off
		4.4.2	Reproduceren fout
		4.4.3	Materiedeskundigheid
	Demotivatie van testers	4.2	Planning, prio, structuur Automatiseren, uitbesteden, testorganisatie
	Omvang van onderhoud	4.4.1	Kick-off Teststrategie
	Invloed van onderhoud op specificaties en testgevallen	4.4.1	Kick-off
		4.4.2	Testbasis door communicatie Reproduceren fout
		4.4.3	Materiedeskundigheid
		4.4.5	Regressietestset aanpassen
	Oneigenlijke taken van testers	4.1	Advies

4 TESTEN IN ONDERHOUD

In het vorige hoofdstuk zijn een aantal aandachtspunten genoemd met betrekking tot het testen in onderhoud. In dit hoofdstuk worden hiervoor, in de diverse paragrafen, oplossingen, tips, hints en aanvullende activiteiten aangedragen.
Aangezien testen in onderhoud niet altijd in projectverband plaatsvindt zijn er testaspecten te benoemen die projectoverstijgend ingericht kunnen worden.
In de eerste paragrafen wordt daarom aandacht besteed aan de inrichting van:

- een standaard testproces;
- de testorganisatie;
- de testinfrastructuur.

Vervolgens worden de verschillen met het standaard testproces in het TMap faseringsmodel ("wyber") toegelicht [Aalst-2 2006]. Hierin wordt vooral aandacht besteed aan de planbare onderhoudssoorten, omdat dat nu eenmaal de meest voorkomende vorm van onderhoud is. Echter ad-hoc, voornamelijk correctief, onderhoud kan hierop afwijkend zijn. Daarom wordt, waar relevant, hier specifiek aandacht aan besteed.

4.1 Standaard testproces

Specifiek proces

In het algemeen kan worden gezegd dat de standaard TMap aanpak integraal toepasbaar is. Onthoud hierbij dat testen een vakgebied blijft met zijn *eigen doel en verantwoordelijkheden*. Als een vermenging van specifieke beheertaken met testtaken gewenst is moet dat weloverwogen gebeuren, omdat anders de kans bestaat dat de testtaken ondergeschikt aan de beheertaken raken. En om te voorkomen dat testen als een geïsoleerde activiteit wordt gezien is het van belang dat de testactiviteiten ook in de wijzigingsprocedure goed uitgedacht en vastgelegd zijn.

Advies

Het uiteindelijke doel van het testen is het komen tot het geven van een *advies ten aanzien van de kwaliteit en risico's*. Om te komen tot een dergelijk advies zijn enkele verschillen en overeenkomsten aan te geven tussen gepland- en ad-hoc onderhoud.
Bij gepland onderhoud moet vooral worden getest of doorgevoerde wijzigingen of nieuwe functionaliteiten goed zijn geïmplementeerd.
Bij ad-hoc onderhoud richt het testen zich erop of de fout hersteld is en of eventuele gevolgschade is verholpen.
Daarnaast geldt voor zowel gepland- als ad-hoc onderhoud dat moet worden getest of er geen (nieuwe) fouten zijn geïntroduceerd. Om dit goed te kunnen doen is het van belang om over een adequate regressietestset te kunnen beschikken. Deze set is extra waardevol als deze is samengesteld op basis van een uitgevoerde risicoanalyse. Als deze set ook nog eens modulair en schaalbaar is opgesteld, dan kan zelfs in een ad-hoc onderhoudssituatie gericht worden getest door het kiezen van de relevante tests, dekkingsgraad en diepgang. In paragraaf 4.4.3 "fase specificatie" is een aanpak beschreven om een dergelijke *schaalbare regressietestset* op te kunnen bouwen. Meer uitleg over hoe bepaald kan worden wat en hoe zwaar er getest moet worden is te vinden onder teststrategie in paragraaf 4.4.1 "fase planning en beheer".

Standaard onderhoudstestplan

Het verdient aanbeveling om niet steeds opnieuw een manier van werken te bedenken. Zowel het testproces als herbruikbare testaspecten zoals testtechnieken, benodigde infrastructuur, organisatie, communicatie, procedures en regressietestset kunnen immers voor een groot deel éénmalig worden bedacht en worden vastgelegd in een *standaard*

onderhoudstestplan[1]. Ook moet hierin al een (beperkte) teststrategiebepaling zijn opgenomen. Door bijvoorbeeld op een systeem een risicoanalyse te doen, kunnen de risicovolle en minder risicovolle delen onderscheiden worden. Een wijziging op een risicovol deelsysteem betekent meer testinspanning dan voor een minder risicovol deelsysteem.
Aangezien de invloed van het kunnen beschikken over een goede teststrategie van grote invloed is op de kwaliteit van het gehele testtraject wordt dit onderwerp in de paragraaf 4.4.1 "fase planning en beheer" nader toegelicht.
Bij ad-hoc correctief onderhoud heeft het oplossen van het productieprobleem de hoogste prioriteit. Hoewel dit weer tot gevolg heeft dat niet alle benodigde stappen van een gestructureerde testaanpak uitgevoerd kunnen worden, is juist dan het aanwezig zijn van een standaard onderhoudstestplan onmisbaar, omdat hierin staat welke testactiviteiten bij ad-hoc onderhoud essentieel zijn en altijd uitgevoerd moeten worden. Als hiernaast ook nog een "eenvoudig" op de teststrategie aan te passen schaalbare regressietestset aanwezig is, dan kan "zelfs" bij ad-hoc correctief onderhoud een kwalitatief goede test worden uitgevoerd.

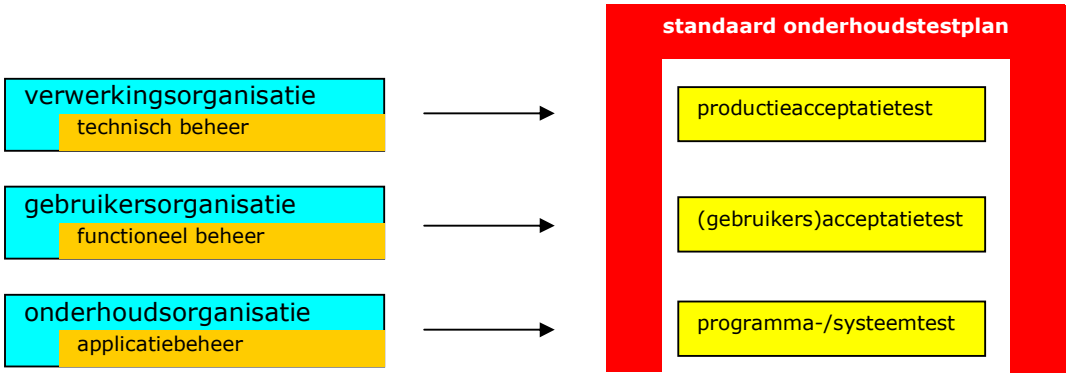
4.2 Testorganisatie

Planning, prioriteiten en structuur beheerorganisaties

In tegenstelling tot een project waarin de aansturing en controle op werkzaamheden vaak wel goed is geregeld (teamleiders) is dit in een beheerorganisatie veel lastiger te realiseren. In een beheerorganisatie kunnen immers meerdere groepen werkzaam zijn die onderhoud aan verschillende systemen of systeemdelen uitvoeren. Hierbij kan het voorkomen dat de controle door de lijnmanager op de uitvoering van testwerkzaamheden niet erg effectief is. Dit zal eerder het geval zijn als voor onderhoud geen aparte testers aanwezig zijn. In die gevallen is het aan te raden binnen elke groep een medewerker, bijvoorbeeld een systeemanalist, als groepsleider aan te stellen en verantwoordelijk te maken voor de kwaliteit van de opgeleverde producten. Deze medewerker controleert dan zijn/haar collega's op het uitvoeren van hun testwerkzaamheden.
Omdat het testen in een onderhoudssituatie vaak niet in projectverband wordt uitgevoerd, zijn medewerkers uit de diverse beheerorganisaties zelden voor 100% van hun tijd beschikbaar voor testwerk. Zodra een productieprobleem optreedt zijn de testwerkzaamheden hier vaak de "dupe" van. Tenslotte verdient het oplossen van het productieprobleem alle aandacht en daarbij vinden beheerders het oplossen van problemen meestal ook leuker dan het uitvoeren van saai en vaak ook nog eens repetitief testwerk. In het algemeen kan worden gezegd dat de "doorsnee" beheerder weinig affiniteit heeft met het testen. Het verdient aanbeveling om vooraf met alle betrokkenen goed door te spreken hoe in een dergelijke situatie gehandeld moet worden. Stel bijvoorbeeld gezamenlijk een *duidelijke planning* op en leg de *prioriteiten* van de medewerkers en hun werkzaamheden vast. Denk hierbij aan het voorbereiden en het uitvoeren van tests in "tandems". Zodra een productieprobleem optreedt wordt dit tandem, tijdelijk, ontbonden. Eén houdt zich bezig met het oplossen van het probleem, de ander blijft zich op het testwerk concentreren.
Uiteraard zal er rekening gehouden moeten worden met de structuur van de beheerorganisaties. Niet elke organisatie heeft deze ingericht volgens de in hoofdstuk 2 "beheer en onderhoud" geschetste indeling.

[1] Een standaard onderhoudstestplan kent diverse verschijningsvormen en naamgevingen. Voorbeelden hiervan zijn: generieke testafspraak (voorbeeldsjabloon aanwezig in bijlage) en generiek mastertestplan.

In het standaard onderhoudstestplan kunnen wel de testsoorten worden benoemd en wie, of welke afdelingen, voor de invulling en uitvoering daarvan verantwoordelijk zijn. In algemene zin kan men zeggen dat de *gebruikersorganisatie* verantwoordelijk is voor de (gebruikers)acceptatietest, de *verwerkingsorganisatie* voor de productie-acceptatietest en de *onderhoudsorganisatie* voor de programma-/systeemtest (zie figuur 2, relatie



beheervormen met testsoorten).

Figuur 2, relatie beheervormen met testsoorten.

Automatiseren, uitbesteden en aparte testorganisatie

Er zijn diverse oplossingen denkbaar om het in de ogen van de beheerder saaie en repetitieve testwerk te minimaliseren. Dit kan bijvoorbeeld door de regressietest te *automatiseren* of door *uitbesteding van het testen*. Vooral het laatstgenoemde wordt door vele organisaties steeds vaker gedaan, al of niet als onderdeel van totale uitbesteding van het onderhoud.
Of, als uitbesteden (nog) een te grote stap is, zijn er beheerorganisaties die besluiten om een *aparte testorganisatie* [Aalst-1 2010] inrichten. Overwegingen hierbij om dit doen zijn onder andere: grootte van de organisatie, risico's die gelopen worden, impact van het probleem of de wijziging, beschikbaarheid van gekwalificeerd personeel, beschikbaarheid van systemen, aanwezige test- en materiekennis, complexiteit van de systemen en verwachte werkaanbod [TestNet 2005].

4.3 Testinfrastructuur

Permanente testomgeving

Iedereen die bekend is met testen weet dat het kunnen beschikken over een bruikbare testinfrastructuur van groot belang is om goed te kunnen testen. Voor systemen die regelmatig aan wijzingen onderhevig zijn is het aan te bevelen een *permanente testomgeving* in te richten. In een dergelijke situatie is het tevens aan te raden om te onderzoeken of het *automatiseren van een regressietest* hier gewenst is. Voor systemen die minder frequent aan wijzigingen onderhevig zijn kan een "*op afroep*" beschikbare testomgeving voldoende zijn.
Overigens geldt voor zowel de permanente- als de "op afroep" testomgeving dat dit er meer dan één kunnen zijn. Denk hierbij aan een testomgeving voor de onderhoudsorganisatie en een acceptatietestomgeving voor de gebruikersorganisatie.

Testomgeving op afroep beschikbaar

Speciale aandacht verdient de situatie van ad-hoc correctief onderhoud. Het komt regelmatig voor dat "patches", na op zijn best enkel een programmatest doorlopen te hebben, in productie worden aangebracht. Toch is het vaak ook mogelijk én raadzaam deze "patch" te testen in bijvoorbeeld een acceptatietest-, trainings- of uitwijkomgeving.

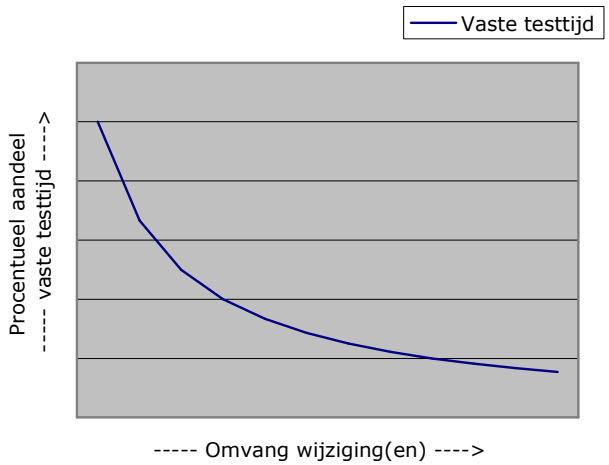
Testdatabase
Een bekend terugkerend discussiepunt is nog altijd of het gebruik van productiebestanden mogelijk, gewenst, noodzakelijk of juist ongewenst is. Om een zo natuurgetrouwe afspiegeling van de productiesituatie te hebben (met name belangrijk als gebruikers gaan testen) is een veel voorkomende werkwijze het kopiëren van de *productiedatabase*[2], waarna deze wordt aangepast en/of verkleind. Nadeel is dat dan niet altijd even eenvoudig te achterhalen is wat de exacte vulling is en dat de database voor ongeveer 95% is gevuld met doorsnee gegevens. Dit in tegenstelling tot een testdatabase die van "scratch" af aan wordt opgebouwd. Dan is exact bekend wat de vulling is en kunnen naast bijvoorbeeld 80% normale gegevens ook 20% "uitzonderingssituaties" worden opgenomen. Complicerende factor kan hierbij zijn het actueel houden van de testdatabase. Denk hierbij aan het draaien van zogenoemde dag-, week-, maandruns, enzovoort.
Welke keuze de beste is zal van geval tot geval variëren.
De keuze wordt nog belangrijker als er geautomatiseerd wordt getest of getest gaat worden. In dat geval is het bezitten van een stabiele bekende uitgangsdatabase vaak onontkoombaar. Dit levert mogelijk een spanningsveld op met het kunnen testen tegen een zo actueel mogelijke database. Dit spanningsveld wordt al wat kleiner als het geautomatiseerd testen zoveel als mogelijk *data onafhankelijk* is c.q. wordt opgezet.

4.4 Testfasering

4.4.1 Fase Planning en beheer

Vaste- en variabele testkosten
Een algemeen "test imago" probleem is dat testen in de perceptie van management altijd te veel tijd in beslag neemt. Bij testen in onderhoud wordt dat nog eens versterkt door het feit dat testen een relatief groot aandeel heeft in het onderhoudstraject. Het is van belang om over het hoe en waarom hiervan duidelijk te zijn naar de opdrachtgever. Hierbij moet beseft worden dat de totale testtijd is opgebouwd uit *vaste- en variabele testtijd en* daarbij dus ook uit vaste- en variabele testkosten. Denk bij vast aan bijvoorbeeld de tijd die nodig is om de testomgeving klaar te zetten of aan het uitvoeren van een "standaard" regressietest en denk bij variabel aan bijvoorbeeld het voorbereiden en testen van doorgevoerde wijzigingen.
Bij het testen van een kleine wijziging is het percentage vaste testtijd hoog. Naarmate de wijzigingen omvangrijker worden neemt het percentage vaste testtijd af. Voorbeeld: Bij het testen van een wijziging wordt altijd een 4 uur durende regressietest uitgevoerd. Als het doorvoeren van een wijziging in het totaal 8 uur duurt bedraagt het percentage vaste testtijd 50% (4/8). Duurt het doorvoeren van één of meer wijzigingen in het totaal 40 uur, dan daalt dit percentage naar 10% (4/40). Zie figuur 3, relatief aandeel vaste testtijd, voor grafische weergave.
Het is van belang om vooraf commitment voor de testaanpak te verkrijgen bij de opdrachtgever van de test, zodat achteraf geen discussies kunnen ontstaan over het relatief grote aandeel van het testen in het onderhoudstraject. Het verzamelen van verhoudingsgetallen tussen het aandeel testen en de overige onderhoudsactiviteiten kan ook een goed inzicht geven aan de opdrachtgever. In het algemeen komt het voor dat het aandeel testen (vast+variabel) zich bevindt tussen 35% - 80% van alle onderhoudsactiviteiten. Hierbij moet in het achterhoofd gehouden worden dat het mogelijk is dat een 80% testaandeel in absolute uren minder kan zijn dan in een onderhoudstraject met een testaandeel van 35%. Dit is vooral afhankelijk van de totale omvang van het onderhoudstraject.

[2] Hierbij moet rekening worden gehouden met de "Wet op bescherming Persoonsgegevens".



Figuur 3, relatief aandeel vaste testtijd.

In ieder geval kan in deze fase veel aan tijd en kwaliteit worden gewonnen indien er een *standaard onderhoudstestplan* klaar ligt en daarmee kunnen ook volgende testfasen sneller en beter worden doorlopen. Eigenlijk wordt hiermee aan één van de grondbeginselen van de TMap fasering vorm gegeven, namelijk het zoveel als mogelijk van het kritieke pad afhalen van testactiviteiten, zodat alleen de testuitvoering zich daar nog op hoeft te bevinden.

Kick-off

Bij het inventariseren van de testbasis wil het nog wel eens voorkomen dat wijzigingen niet goed zijn gedocumenteerd en in het geval van ad-hoc onderhoud dat de concrete aanleiding/oorzaak niet benoemd is. Een beproefde manier om hier meer helderheid in te krijgen is het houden van een *kick-off sessie* met alle relevante partijen (functionele- en technische beheerders, ontwikkelaars, gebruikers, testers). Een impactbepaling, het signaleren van risico's en het opstellen c.q. aanpassen van de teststrategie met tevens aandacht voor *niet-functionele kwaliteitsattributen* zijn onderwerpen die goed passen in een dergelijke kick-off sessie. Houd bij het verzamelen van niet-functionele kwaliteitsattributen rekening met de bestaande situatie. Een aanscherping van bijvoorbeeld een performance eis van drie naar één seconde is meestal lastig te realiseren door middel van onderhoud als daar bij het oorspronkelijke ontwerp van het systeem geen rekening mee is gehouden.
Specifiek in het geval van ad-hoc onderhoud kan tijdens een kick-off sessie ter sprake komen hoe een fout in een testsituatie gereproduceerd kan worden.
De uitdaging bij het organiseren van een kick-off sessie bestaat uit het afstemmen van de agenda's van de gewenste deelnemers versus de beschikbare, meestal beperkte, testdoorlooptijd.

Releasematig opleveren

Een ander verschijnsel dat ook regelmatig voorkomt bij testen in onderhoud is de verspreide oplevering van deeloplossingen aan de testers. Bijvoorbeeld in de vorm van losse programma's of deelsystemen. Hierdoor worden testers gedwongen een gedetailleerde administratie te voeren om te kunnen overzien welke deeloplossingen samen genomen moeten worden om een bepaalde wijziging te kunnen testen. Dit vergroot de kans op fouten en leidt mogelijk ook tot een groot aantal hertests. Beter is het om vooraf met de onderhoudsorganisatie afspraken te maken over de wijze van oplevering. Een *releasematige oplevering* is hier een goede oplossing. Tevens wordt hiermee de vaste testtijd teruggebracht omdat de uitvoering van de regressietest éénmalig voor de hele release kan plaatsvinden.
Niet altijd wordt onderhoud in de lijn uitgevoerd. Overwegingen om onderhoud in een *project* onder te brengen zijn bijvoorbeeld de hoeveelheid geld, tijd en resources die er

mee gemoeid zijn. Ook het aantal betrokken disciplines/rollen en de eventuele risico's die een wijziging met zich meebrengt spelen een rol. Door onderhoud op te nemen in een project kan planmatiger worden gewerkt, worden er duidelijke afspraken gemaakt over budgetten en worden beslissingen op managementniveau genomen.

Teststrategie

Het standaard onderhoudstestplan moet definiëren hoe de teststrategie in onderhoud opgesteld wordt. In het standaard onderhoudstestplan kan een risiconiveau aan systeemdelen worden toegekend (bijvoorbeeld op basis van complexiteit en bedrijfsbelang van het systeemdeel). Vervolgens wordt voor elke wijzigingsopdracht een risicoanalyse uitgevoerd op basis van de complexiteit van de wijziging en de ervaring van de ontwikkelaar. Als niet duidelijk is waar het systeem precies is gewijzigd kan de nieuwe broncode met de oude worden vergeleken om aanpassingen in de code te detecteren. Aangezien er een (test)verschil is tussen nieuwbouw en onderhoud heeft dit gevolgen voor de uit te voeren stappen van de risicoanalyse.
Het voornaamste verschil tussen nieuwbouw en onderhoud voor de teststrategie is de *foutkans*. Er wordt bij onderhoud een aantal aanpassingen, meestal naar aanleiding van probleemrapporten of wijzigingsvoorstellen, gedaan op een al bestaand systeem. Deze wijzigingen kunnen foutief geïmplementeerd zijn en dienen te worden getest. Bij het aanpassen is er ook een kleine kans dat er onbedoeld fouten in ongewijzigde delen van het systeem worden geïntroduceerd, waardoor het systeem in kwaliteit achteruit gaat. Dit verschijnsel van kwaliteitsachteruitgang heet regressie en is de reden dat ook de ongewijzigde delen van het systeem getest worden.
Deze verdeling van foutkansen bij onderhoud betekent voor de strategiebepaling dat vooral de risicoclassificatie van de deelsystemen anders komen te liggen dan bij een nieuwbouwsituatie. Een deelsysteem dat bij de nieuwbouw een hoog risico had, blijft bij de onderhoudsrelease misschien wel ongewijzigd. Omdat de kans op regressie dan het enige risico is, hoeft er veel minder grondig getest te worden. Om die reden kan voor onderhoud de strategiebepaling voor een testsoort worden aangepast door in de stappen van de strategiebepaling het begrip "deelobjecten" te vervangen door het begrip "wijzigingen". Een dergelijke vorm van strategiebepaling wordt ook wel (test-)impactanalyse genoemd. Per wijziging (een geaccepteerd wijzigingsvoorstel of een opgelost probleemrapport) wordt geïnventariseerd welke delen van het systeem worden gewijzigd, welke delen van het systeem mogelijk worden beïnvloed door de wijziging en welke kwaliteitsattributen relevant zijn. Er zijn diverse variaties mogelijk voor het testen van elke wijziging, geheel afhankelijk van de gelopen risico's:

- een beperkte test, alleen gericht op de wijziging;
- een volledige (her-)test van de functie waarin de wijziging is aangebracht;
- het testen van de samenhang tussen de gewijzigde functie en de functies er direct omheen;
- het gehele systeem testen.

Tevens wordt de regressietest van het systeem als geheel onderkend. De regressietest richt zich voornamelijk op de samenhang tussen de gewijzigde en ongewijzigde delen van het systeem, omdat hier de kans op regressie het grootst is. Indien de strategiebepaling voor de nieuwbouw beschikbaar is, kunnen de hier toegekende belangen aan deelobjecten een rol spelen bij de samenstelling van deze regressietest. Een regressietest kan beperkt of volledig worden uitgevoerd, afhankelijk van de risico's én van de benodigde testinspanning. Bij het uitvoeren van regressietests komt het gebruik van testtools het best tot zijn recht. Het grote voordeel van automatisering van de regressietest is dat tegen een geringe inspanning elke keer de volledige test kan worden uitgevoerd en geen keuze hoeft te worden gemaakt welk deel van de regressietest wel of niet uitgevoerd gaat worden.
De keuze voor het formuleren van de strategie in termen van deelobjecten of wijzigingsvoorstellen wordt beïnvloed door het aantal wijzigingsvoorstellen en het deel van het systeem dat geraakt wordt door de wijzigingen. Hoe meer wijzigingen en hoe groter het deel van het systeem dat geraakt wordt, hoe meer de voorkeur uitgaat naar

het bepalen van de teststrategie op het niveau van deelobjecten in plaats van wijzigingsvoorstellen.
De stappen van de strategiebepaling en begroting zijn bij de keuze voor wijzigingsvoorstellen als volgt:

1. Verzamelen wijzigingen (in de vorm van geaccepteerde wijzigingsvoorstellen en opgeloste probleemrapporten);
2. Bepalen impact (analyse van geraakte systeemdelen) per wijziging;
3. Bepalen schade bij falen (hoog, midden, laag) per wijziging;
4. Bepalen kans op falen (hoog, midden, laag) per wijziging;
5. Bepalen risicoklasse (A, B, C) per wijziging [Baarda 2004];
6. Wijzigingen groeperen per risicoklasse en hier regressietest aan toevoegen;
7. Bepalen omvang per wijziging (en regressietest);
8. Bepalen testinspanning per wijziging op basis van omvang en risicoklasse van desbetreffende wijziging (en regressietest).

Een voorbeeld van een "strategietabel" na stap 6:

	Risicoklasse	**Omvang**	**Testinspanning**
Wijzigingsvoorstel 1	A		
Wijzigingsvoorstel 4	A		
Regressietest totale systeem	A		
Opgelost probleem rapport 2	B		
Wijzigingsvoorstel 3	B		
Opgelost probleemrapport 1	C		
Wijzigingsvoorstel 2	C		

Omvang (ontwerp/bouw) en Testinspanning in uren.

In bovenstaand voorbeeld is aan de regressietest risicoklasse A toegekend. Dit betekent dat er veel waarde wordt gehecht aan het uitvoeren van een regressietest. De totale inspanning voor een regressietest is vaak veel groter dan de testinspanning die nodig is voor het gedetailleerd testen van de wijzigingen. De reden hiervoor is dat in onderhoud meestal maar een zeer beperkt aantal functies wijzigt en de regressietest het gehele systeem bestrijkt. Met behulp van de schaalbare regressietestset (zie paragraaf fase 4.4.3 "specificatie") kan voor een andere strategie worden gekozen, bijvoorbeeld voor een regressietestset te selecteren met een lagere dekkingsgraad, lagere diepgang, enzovoort.
Het grootste deel van het onderhoud is planbaar en de beschreven strategiebepaling is hierbij zonder meer toepasbaar. Lastiger wordt het bij ad-hoc onderhoud, waarbij een productiestoring verholpen moet worden en het systeem zo snel mogelijk weer in productie moet. Een formele strategiebepaling kost hier vaak teveel tijd. Wel kunnen van *tevoren* een aantal strategiescenario's voorbereid worden: als programma X stuk loopt en hersteld wordt, wat zou dan getest moeten worden? Enkel de wijziging of ook een (beperkte) regressietest? Deze scenario's ondersteunen een zo goed mogelijke test bij ad-hoc onderhoud.

4.4.2 Fase Voorbereiding

Testbasis door communicatie
In deze fase wordt een detailintake uitgevoerd op de foutmelding of het wijzigingsvoorstel. Zoals eerder gezegd komt het regelmatig voor dat er onvoldoende documentatie is om er testgevallen tegen te kunnen schrijven. In de kick-off sessie kan hier meer duidelijkheid in worden gebracht. Maar beter is het om dit al eerder geregeld te hebben, bijvoorbeeld door *vooraf te communiceren* wat de testers in bijvoorbeeld een

wijzigingsvoorstel verwachten aan te treffen. Dit leidt op den duur tot een meer structurele aanpak van het documentatieprobleem dan het bespreken in een kick-off sessie. Aangezien het onderhoud op een bestaand systeem betreft is het vaak ook mogelijk en gewenst om het systeem zelf als testbasis (referentie) te gebruiken.

Reproduceren oorspronkelijke fout
Naast het testen van wijzigingsvoorstellen wordt ook getest of een probleem goed is opgelost. Om dat laatste adequaat te kunnen doen, zal eerst uitgezocht moeten worden wat het probleem nu eigenlijk was. In deze fase moet dan ook getracht worden de *fout te reproduceren*. Dit is feitelijk de enige manier om, na implementatie van de oplossing ("patch"), in de fase testuitvoering vast te kunnen stellen of de fout daadwerkelijk is opgelost. Het is raadzaam om hierbij met de bouwer samen te werken. Deze staat immers voor dezelfde opgave.

4.4.3 Fase Specificatie

Materiedeskundigheid
Optimaal is het als vanuit het nieuwbouwproject de regressietestset aan beheer wordt overgedragen. In het project is hiervoor tijd ingepland en is materiedeskundigheid bijeen gebracht. Als dit niet gebeurd is en later in de onderhoudssituatie alsnog een regressietestset moet worden samengesteld, dan loopt dit vaak stuk op de beschikbare tijd, materiekennis en motivatie. Het opleveren van een regressietestset door het project moet een acceptatiecriterium zijn van de beheerorganisaties.
Bij het specificeren van onderhoud specifieke testgevallen moeten minimaal de beginsituatie, de actie en het verwachte resultaat worden vermeld. In de fase specificatie worden bij voorkeur "slechts" de aanvullende testgevallen geschreven of bestaande testgevallen aangepast voor het testen van de wijzigingen. Lastig punt hierbij is dat er meestal weinig tijd en materiedeskundigheid aanwezig is om testgevallen te specificeren en dat er weinig kennis van testspecificatietechnieken aanwezig is. Het verdient daarom aanbeveling om tijdens de kick-off sessie zoveel mogelijk kennis van de wijziging te vergaren en om er voor te zorgen dat tijdens deze specificatiefase de inbreng van gebruikers is geregeld. Naast het specificeren van aanvullende testgevallen moet bekeken worden of de wijzigingen gevolgen hebben voor de regressietestset en moet deze zo nodig aangepast worden. Hierbij verdient het aanbeveling om als uitgangssituatie te kunnen beschikken over een *schaalbare regressietestset*. Tijdens de kick-off en/of teststrategie sessie is al bepaald of de hele set of bepaalde testgevallen uit deze set uitgevoerd moeten worden. Hoe een schaalbare regressietestset opgebouwd en gebruikt kan worden, wordt in de volgende paragraaf toegelicht.

Schaalbare regressietestset
Een goede regressietestset is van onschatbare waarde. Zeker in de beheersituatie is ze een archief van kennis over de toepassing en een bron van kant en klare testgevallen voor regressietests.
In de praktijk zijn deze testsets vaak gebrekkig. Het opzetten van een doordachte regressietestset is dan ook eenvoudiger gezegd dan gedaan.
In deze paragraaf wordt een aanpak beschreven voor het opbouwen, gebruiken en beheren van regressietestsets gebaseerd op het door Sogeti ontwikkelde TestKubus principe [TestKubus 2005]. Daarin worden een aantal samenhangende principes beschreven waarmee het mogelijk wordt om:
- testgevallen te specificeren en uit te voeren op basis van prioritering;
- snel en adequaat te rapporteren over de voortgang van testspecificatie en/of testuitvoering;
- testtrajecten nauwkeurig te plannen en te begroten;
- snel en variabel regressietestsets samen te stellen;
- wijzigingen in het testobject eenvoudig te verwerken in de testset.

De TestKubus is in concept weinig meer dan een verzameling aanvullende gegevens die per testgeval worden vastgelegd: de testgevallen in de testset worden 'geclassificeerd'. Met behulp van deze classificaties kunnen langs allerlei dwarsdoorsneden subsets van testgevallen uit de testset worden geselecteerd.
Voorbeelden van classificaties zijn:

- applicatie
- deelobject
- functie
- productrisico[3]
- risicoklasse
- proces(onderdeel)
- release
- requirement
- transactie
- zwaartecategorie

Een goede selectie van classificaties en het correct classificeren van de testgevallen is bepalend voor de bruikbaarheid van dit concept.
Essentieel in de onderhoudssituatie is de classificatie naar zwaartecategorie. Deze classificatie geeft het 'gewicht' van het testgeval in de testset aan. In het bijzonder deze classificatie maakt het mogelijk om met een variabele diepgang risicogebaseerd te regressietesten.
De toepassing van deze zwaartecategorieën, bijvoorbeeld bij het samenstellen van regressietestsets, is als volgt (bij indeling in drie categorieën):

- Door van een deelobject alleen de testgevallen van categorie 1 te selecteren ontstaat een kleine regressietestset. Deze subset wordt gebruikt voor een deelobject waarop geen aanpassingen zijn gedaan (of voor een intaketest op een nieuw of ingrijpend gewijzigd deelobject).
- De testgevallen van categorie 2 (= inclusief categorie 1) leveren een normale regressietestset op, bijvoorbeeld voor een deelobject waarin aanpassingen zijn gedaan.
- De testgevallen van categorie 3 (= inclusief categorie 1 en 2) dekken het totale deelobject af en worden toegepast bij nieuwe of ingrijpende gewijzigde deelobjecten.

Het concept schrijft geen vaste testspecificatietechnieken voor. Alle bestaande technieken kunnen worden gebruikt om de testgevallen in de testset te specificeren. De keuze voor een techniek wordt op de normale manier bepaald aan de hand van de teststrategie.
Ook aan de mate van detail waarin de testgevallen gespecificeerd worden, worden geen eisen gesteld. Wanneer wordt voorzien dat testgevallen uitgevoerd gaan worden door testers zonder materiekennis, kunnen de testgevallen meer in detail worden uitgeschreven.
Slechts op één punt stelt het concept een eigen specifieke eis aan de testgevallen. Ze moeten namelijk onafhankelijk van elkaar zijn. Dit heet het zogenaamde onafhankelijkheidsprincipe van het concept.

- Er mogen geen afhankelijkheden bestaan tussen testgevallen.
  Wanneer de uitvoering van het ene testgeval voorwaardelijk is voor de uitvoering van het andere, ontstaan problemen bij het samenstellen van testsets uit de regressietestset. Hierdoor kan het voorkomen dat er eerst allerlei testgevallen uitgevoerd moeten worden die niet tot de gekozen testset behoren.
- Testgevallen moeten parallel aan elkaar uitgevoerd kunnen worden.
  Testgevallen die voor een bepaalde periode exclusief gebruik van de testomgeving vereisen, belemmeren de uitvoering van andere testgevallen. Dit bemoeilijkt het plannen van de doorlooptijd van het testtraject.

[3] Voor het bepalen c.q. het classificeren van de productrisico's is het aan te bevelen de werkwijze te volgen zoals deze wordt beschreven in de business driven testmanagement aanpak [Baarda 2004]

Bij toepassing van dit concept worden de omvang van de (regressie)testset en de daarmee samenhangende activiteiten in het testtraject concreet meetbaar. Door per testgeval enkele statussen bij te houden kan op elk gewenst moment een actueel beeld van de voortgang worden gegeven. Wanneer metrics bijgehouden worden van afgeronde of lopende testtrajecten, kan hiermee een nauwkeurige planning opgesteld worden voor toekomstige trajecten of voor het vervolg van lopende activiteiten.

Voor een nadere toelichting wordt naar het desbetreffende white paper verwezen [TestKubus 2005].

4.4.4 Fase Uitvoering

De activiteiten die in deze fase uitgevoerd moeten worden zijn voor testen in onderhoud niet anders dan in nieuwbouwsituaties.
Hier worden de aangepaste software en de eventuele herstelprogramma's getest.

Testkwaliteitverbetering bij ad-hoc onderhoud
Ten aanzien van ad-hoc correctief onderhoud lijkt het "op het oog" dat de testfase testuitvoering de enige testfase is die wordt uitgevoerd, omdat alle testactiviteiten in veel gevallen in één keer worden uitgevoerd. Dit heeft uiteraard te maken met storingen die onmiddellijk om een oplossing vragen. Hierbij valt te denken aan een productierun die 's avonds laat dumpt, een netwerk met een paar honderd gebruikers die "hangen", een mailing met foute adressen, enzovoort. Bij het oplossen van dit soort problemen gelden andere wetten, andere procedures. Zoals eerder gezegd, zal het niet mogelijk zijn de benodigde stappen van een gestructureerde testaanpak uit te voeren.
Maar door uit de *schaalbare regressietestset*, op basis van de in het *standaard onderhoudstestplan* reeds *aanwezige risicoanalyse*, die relevante testgevallen te selecteren die betrekking hebben op het probleem bestaat de mogelijkheid om toch (snel) een beperkte regressietest uit te kunnen voeren. Naast het testen of de fout is verholpen kan dan ook met minimale inspanning getest worden of er geen nieuwe fouten zijn geïntroduceerd. Overigens kan men bij een laag risico besluiten om achteraf (bijvoorbeeld de volgende dag) te testen.
Ook ten aanzien van ad-hoc onderhoud is het dus mogelijk om door middel van een bepaalde testaanpak, waarbij het gewenst is dat alle testfasen (hoe minimaal dan ook) worden doorlopen, een kwaliteitsverbetering te realiseren. Belangrijk is dat er al een goede risicoanalyse met betrekking tot het informatiesysteem uitgevoerd moet zijn en dat de resultaten daarvan in het standaard onderhoudsplan moeten zijn opgenomen. Tevens moeten op basis van deze resultaten de juiste classificaties en waarden in de schaalbare regressietestset zijn aangebracht.

4.4.5 Fase Afronding

Regressietestset aanpassen
Het actueel houden van een regressietestset wil nog wel eens "vergeten" worden. Het is daarom aan te bevelen om deze activiteiten *standaard* in deze testfase op te nemen. Tijdens de testuitvoering kan het zijn voorgekomen dat het systeem toch anders reageerde dan in het testgeval werd aangenomen. Als deze aanname fout was, moet het testgeval conform de productiesituatie worden aangepast. Verder moet bepaald worden of er, en zo ja welke, nieuwe testgevallen aan de bestaande regressietestset moeten worden toegevoegd. Dit kan éénvoudig door hiervoor de wijzigingen en eventuele foutmeldingen als basis te gebruiken. Let hierbij ook op het opnemen van de juiste classificaties ten behoeve van de schaalbaarheid van de regressietestset.
Naast testgevallen die resulteren uit geplande onderhoudstrajecten, komen er mogelijk ook *testgevallen uit ad-hoc correctief onderhoud*. Hoewel in het laatste geval vaak wordt gezegd dat het "slechts" het oplossen van een probleem betreft, waarbij de functionaliteit ongewijzigd blijft en het daarom niet nodig zou een specifiek hiervoor gemaakt testgeval

in de regressietestset op te nemen, kan het om de volgende reden toch wel nuttig zijn om dat te doen.
Het probleem is namelijk in één bepaalde software versie opgelost, maar de wijziging moet ook in de volgende softwareversies worden doorgevoerd. Het gebeurt regelmatig dat dit, om wat voor reden dan ook, niet gebeurt en daarom is het verstandig om daarvoor toch een specifiek testgeval in de regressietestset opgenomen te hebben.
Tot slot kan het gewenst zijn de risicoanalyse in het standaard onderhoudtestplan aan te passen. Dit heeft met name betrekking op de faalkans van de geteste systeemdelen welke op basis van het aantal gevonden bevindingen aangepast kunnen worden.

5 LITERATUUR

- [Aalst-1 2010]
  Aalst, L. van der, (2010), *TSite®, een overview. Een permanente testorganisatie*, Diemen, versie 2.1, Sogeti white paper

- [Aalst-2 2006]
  Aalst, L. van der, Broekman, B., Koomen, T., Vroon, M. (2006), *TMap Next, voor resultaatgericht testen*, 's-Hertogenbosch: Uitgeverij Tutein Nolthenius, ISBN 90-72194-79-9

- [Delen 1992]
  Delen, G.P.A.J., Looijen, M., (1992), *Beheer van de informatievoorziening*, Rijswijk, Cap Gemini Publishing

- [Grady 1987]
  Grady, R., (1987), *Software Metrics: Establishing a Company-Wide Program*, Prentice-Hall

- [Baarda 2004]
  Baarda, R., Koomen, T., (2004), *TMap® Test Topics* (hoofdstuk 4), Tutein Nolthenius, 's-Hertogenbosch, 1ste druk, ISBN 90-72194-70-5

- [Lint 2005]
  Lint, P. van., Kortman, M., (2005), *Sjabloon Generieke Testafspraak*, Rotterdam, versie 2.0, Sjabloon ontstaan uit samenwerking van Sogeti met een klant

- [Tamai 1992]
  Tamai, T., Torimitsu, Y., (1992*), Software Lifetime and its Evolution Process over Generation*, Proceedings of 1992 Conference on Software Maintenance

- [TestKubus 2005]
  Delprat, H., Jelgerhuis, R., Suiveer, M., Verweij, A., Wester, G., Winckelmann, P. von (2005), *De TestKubus, aanpak voor het structureren en beheren van testsets*, Diemen, versie 1.0, Sogeti white paper

- [TestNet 2005]
  Thema-avond TestNet, (2005), *Testen in onderhoudssituatie*, 23 februari 2005 te NBC Nieuwegein

6 BIJLAGE: SJABLOON GENERIEKE TESTAFSPRAAK

Deze bijlage bevat een sjabloon voor het opstellen van de *Generieke TestAfspraak* (GTA) [Lint 2005].

De GTA is de afspraak tussen opdrachtgever en opdrachtnemer betreffende de samenwerking op het gebied van testen tijdens regulier onderhoud.
Opdrachtgever en opdrachtnemer hebben beide behoefte aan afspraken met betrekking tot testen waardoor misverstanden betreffende de kwaliteit bij oplevering worden voorkomen. Het is echter inefficiënt om voor elke opdracht telkens opnieuw vooraf afspraken te maken. Daarom is ervoor gekozen algemene afspraken op het gebied van testen te maken in de vorm van een GTA. Deze is leidend voor de tests die tijdens alle onderhoudsopdrachten voor de opdrachtgever in een bepaald kader (bijvoorbeeld één of meerdere systemen) worden uitgevoerd. Mocht het in voorkomende gevallen niet mogelijk of wenselijk zijn de GTA te volgen, dan wordt een specifieke testafspraak opgesteld.

Het doel van de GTA is dat duidelijkheid over de principiële keuzes in het testproces bestaat tussen beide partijen en naar derde partijen toe. Bij dit laatste moet gedacht worden aan klanten van de opdrachtgever, controlerende en auditerende instanties, et cetera.
Zolang de GTA van kracht is, wordt deze jaarlijks herijkt en bekrachtigd.

De GTA omvat de volgende informatie:
1. Inleiding
2. Teststrategie
3. Organisatie
4. Testproducten
5. Infrastructuur
6. Akkoord

<< In dit sjabloon staat tekst tussen twee maal 2 haken (<< Tekst >>). Deze tekst bevat commentaar of uitleg en dient te worden verwijderd of vervangen door de definitieve tekst. Alle overige standaard tekst mag naar behoefte gewijzigd worden.>>

1. Inleiding

1.1 Datum

Ingangsdatum: << Ingangsdatum >>
Geldig tot en met: << Laatste datum geldigheid >>

1.2 Partijen

Opdrachtgever: << Opdrachtgever >>
Opdrachtnemer: << Afdeling >>

1.3 Randvoorwaarden en uitgangspunten

<< Aan welke randvoorwaarden moet voldaan zijn en welke uitgangspunten gelden voor een goede uitvoering van de GTA. >>
Randvoorwaarden:

•

Uitgangspunten:

•

1.4 Scope

<< Geef hier een beschrijving en zo mogelijk een afbeelding van alle systemen, applicaties en processen die onder deze GTA vallen. Neem indien mogelijk over uit het testplan van het project dat naar beheer wordt overgedragen. >>

2. Teststrategie

<< Ten behoeve van de GTA voor een beheersituatie wordt éénmalig een teststrategie gemaakt en in dit hoofdstuk opgenomen.
Door de teststrategie in de GTA is voor onderhoud altijd een basis risico-inschatting aanwezig. De teststrategie voor een onderhoudsopdracht kan nu snel bepaald worden door de complexiteit van de wijziging zelf mee te nemen.>>
<< Bij overdracht van een project naar beheer wordt vóór het einde van het project de teststrategie uit het testplan in de GTA overgenomen. Ook kan de teststrategie uit een testplan aan het einde van een project aan een al bestaande GTA worden toegevoegd. De nieuwe of aangepaste GTA is een maintenance product van het project en moet als zodanig in de PID van het project benoemd zijn. >>

2.1 Systeemdelen

<< Splits de scope van deze GTA (par. 1.4) uit naar bijv. applicaties en systeemdelen. >>

2.2 Risicotaxatie

<< Geef hieronder per applicatie/systeemdeel de uitkomsten weer van de risicotaxatie. De risicotaxatie wordt uitgevoerd volgens TMap. Bij de risicotaxatie moet voor elke systeemdeel bedacht worden welke kans op fouten bestaat door het uitvoeren van een onderhoudsopdracht en wat de gevolgen kunnen zijn. Neem indien mogelijk teststrategieën over uit testplannen. >>

Applicatie 1					
Systeemdeel	**Faalkans**		**Schade**		**Risico**
Systeemdeel 1	L				
Systeemdeel 2	M				
Systeemdeel 3	H				
…					

Applicatie 2					
Systeemdeel	**Faalkans**		**Schade**		**Risico**
Systeemdeel 1	L				
Systeemdeel 2	M				
Systeemdeel 3	H				
…					

<< Naast deze risicotaxatie in de GTA moet ook binnen elke onderhoudsopdracht nog een risicotaxatie worden uitgevoerd. Met als uitgangspunt het hierboven vermelde risiconiveau wordt ingeschat wat de invloed van de complexiteit van de wijziging hierop is.>>
<< Verder is het denkbaar dat in de GTA geen risicotaxatie wordt opgenomen maar dat deze voor elke onderhoudsopdracht wordt uitgevoerd. >>

2.3 Kwaliteitsattributen en testsoorten

<< Geef hier aan welke kwaliteitsattributen belangrijk zijn voor de diverse applicaties/systeemdelen. Bepaal welke testsoorten nodig zijn tijdens onderhoud om de diverse kwaliteitsattributen te testen.
Neem indien mogelijk passages over kwaliteitsattributen en testsoorten over uit het testplan. >>

2.4 Testspecificatie(technieken)

<< Beschrijf hier hoe de tests in de diverse testsoorten worden gespecificeerd. Geef indien van toepassing de te gebruiken testspecificatietechnieken aan. Neem indien mogelijk over uit het testplan.
Geef ook aan dat zoveel mogelijk gebruik gemaakt wordt van bestaande logische testspecificaties (bijv. een regressietestset) die zijn overgedragen vanuit het project of zijn ontwikkeld/aangepast in de beheersituatie. >>

2.5 Testbasis

<< Op basis van welke documentatie worden de tests gespecificeerd, uitgevoerd en gecontroleerd? >>

Nr.	Naam	Versie	Datum
1.	<< Document >>	x.x	dd-mm-jjjj
2.	<< Document >>	x.x	dd-mm-jjjj
3.	<< Document >>	x.x	dd-mm-jjjj

3. Organisatie

Dit hoofdstuk beschrijft de testorganisatie, het beheer en de communicatie met betrekking tot het testen tijdens de onderhoudsopdrachten.

3.1 Organisatiestructuur

<< Geef indien bekend de organisatiestructuur van de testorganisatie. Indien sprake is van een vaste bezetting van testrollen of testfuncties, geef deze dan ook hier aan. >>
<< Welke escalatieniveaus zijn gedefinieerd? >>

3.2 Testprocesbeheer

<< Beschrijf hoe en door wie de voortgang en de budget- en tijdbesteding worden geregistreerd en bewaakt. Beschrijf welke kwaliteitsindicatoren en teststatistieken worden geregistreerd, door wie dit gebeurt en hoe hierover wordt gerapporteerd. Beschrijf waar en door wie afwijkingen (specifieke testafspraken) op deze GTA worden geregistreerd. >>

3.3 Bevindingenbeheer

<< Beschrijf voor welke testsoorten tijdens de uitvoering van onderhoudsopdrachten bevindingen worden geregistreerd in de bevindingenadministratie en wie dit uitvoert. Verwijs naar een bestaande bevindingenprocedure en -administratie. Geef aan wie verantwoordelijk is voor de analyse van bevindingen en hoe de rapportage over bevindingen plaatsvindt. >>

3.4 Rapportage

<< Beschrijf wanneer testverslagen en voortgangsrapportages gemaakt moeten worden en welke informatie deze minimaal moeten bevatten. Beschrijf ook hoe en door wie een vrijgaveadvies wordt gegeven. >>

3.5 Overleg

<< Geef aan wanneer en met welke betrokkenen testoverleg plaatsvindt. >>

3.6 Vrijgave

<< Hoe en door wie vindt vrijgave voor productie plaats. >>

3.7 Borging

<< Geef aan hoe borging van het testproces plaatsvindt. Wat wordt in bijvoorbeeld een opdrachtbesturingssysteem vastgelegd (bijv: getaxeerd risiconiveau, uitgevoerde tests, vrijgave)? >>

<< Wie beheert de GTA? >>

4. Testproducten

In dit hoofdstuk wordt beschreven welke testproducten tijdens onderhoudsopdrachten worden vervaardigd en beheerd. Bij het opstellen van testproducten wordt zo veel mogelijk gebruik gemaakt van bestaande templates.

4.1 Testware

<< Testware is alle testdocumentatie die tijdens het testproces wordt geproduceerd en voor onderhoudsdoeleinden kan worden hergebruikt en daarom overdraagbaar en onderhoudbaar moet zijn. Bij regressietesten wordt bijvoorbeeld vaak gebruik gemaakt van bestaande testware. Voorbeelden van testware zijn logische testspecificaties en testdraaiboeken.
In deze paragraaf wordt beschreven welke testware bestaat (bijv. vanuit projecten overgedragen) tijdens onderhoudsopdrachten onderhouden dient te worden. >>

<< Onderstaande tabel kan gebruikt worden. Het is echter mogelijk dat een andere notatie gekozen wordt die beter bij een specifieke beheersituatie past.>>

Testware					
Type testproduct	**Systeemdeel**	**id./doc. nr.**	**Versie**	**Datum**	**Opgeslagen in**

4.2 Overige Testdocumentatie

<< Tijdens het testproces worden diverse documenten ontvangen of zelf opgesteld die betrekking hebben op testen, zoals: projectplannen, overlegverslagen, memo's en rapportages. Geef hier een overzicht van deze documenten. >>

<< Onderstaande tabel kan gebruikt worden. Het is echter mogelijk dat een andere notatie gekozen wordt die beter bij een specifieke beheersituatie past.>>

Overige testdocumentatie					
Type document	**Systeemdeel**	**id./doc. nr.**	**Versie**	**Datum**	**Opgeslagen in**

4.3 Testproductenbeheer

<< Beschrijf hier hoe en door wie het beheer van de testproducten plaatsvindt. Welke documenten worden wel en niet bewaard? >>

5. Infrastructuur

Het hoofdstuk Infrastructuur geeft een beschrijving van de testomgeving, de gebruikte testtools, de benodigde testdata en het beheer hierop.

5.1 Testomgeving

<< Beschrijf hier de omgevingen die t.b.v. ontwikkeling, tests, acceptatietests en productie ter beschikking staan aan beheer en aan de klant. Geef aan welke omgevingen voor welke testsoorten bedoeld zijn. Een voorbeeld hiervan is:

- Ontwikkel/testomgeving: unittest en systeemtest;
- Acceptatieomgeving: FAT en GAT;
- Productieomgeving: hierin worden geen tests uitgevoerd.

Doe dit indien nodig voor alle applicaties/systeemdelen. >>

<< Beschrijf van welke partijen het testteam afhankelijk is bij het opbouwen en onderhouden van de test- en acceptatieomgevingen en de afspraken die hieromtrent gemaakt zijn. >>

<< Beschrijf hoe de overdracht van software tussen de omgevingen plaatsvindt: wie is verantwoordelijk voor welke omgeving en geeft fiat voor installatie. >>

5.2 Testtools

<< Beschrijf welke testtools beheer en de klant ter beschikking staan. Geef aan of deze gekoppeld zijn aan specifieke testomgevingen. Hierbij kan bijvoorbeeld gedacht worden aan tools voor het verzamelen van testgegevens, een tool voor het registreren van bevindingen en een tool voor geautomatiseerde testuitvoering. >>

5.3 Testdata

<< Beschrijf de methode(n) en bron(nen) om tot testgegevens (databasestanden, berichten, etc.) te komen en vermeld de verantwoordelijke partij. >>

<< Beschrijf hoe en door wie de testgegevens in de diverse omgevingen worden geïnstalleerd. >>

5.4 Overige testinfrastructuur

<< Indien andere infrastructurele voorzieningen (zoals bijvoorbeeld testwerkplekken, PC's, netwerken of telefoons) tot de testinfrastructuur gerekend worden, beschrijf deze dan ook hier. Geef aan welke partij verantwoordelijk is voor het totstandkomen van deze voorzieningen en welke afspraken hierover zijn gemaakt. >>

5.5 Beheer testinfrastructuur

<< Geef aan wie verantwoordelijk is voor het beheer van de omgevingen en welke back-up en restore procedures bestaan. Geef ook aan wie de testtools, testdata en overige testinfrastructuur beheert. >>

6. Akkoord

6.1 Evaluatie

Opdrachtgever en opdrachtnemer spreken af dat de GTA en de bijlagen tenminste éénmaal per jaar, voorafgaand aan het einde van de geldigheidstermijn, geëvalueerd worden. Indien nodig worden tussentijdse evaluaties afgesproken.

6.2 Bijlagen

Nr.	Naam	Versie	Datum
1.	<< Bijlage >>	x.x	dd-mm-jjjj

2.	<< Bijlage >>	x.x	dd-mm-jjjj
3.	<< Bijlage >>	x.x	dd-mm-jjjj

6.3 Accordering

Namens << Opdrachtgever >>	Namens << Beheer >>
<< Naam >>	<< Naam >>
Datum:	Datum: