ICT GROUP Improve Quality Services

WHITEPAPER

Sneller en beter testen met **Model-Based Testing**, maar... Selecteer je testcases **zorgvuldig**

Rond het jaar 2010 bevond Model-Based Testing (MBT) zich op de top van de hype cycle. Het werd gezien als een must-have. De methode zou relatief simpel ingevoerd kunnen worden met beperkte investeringen. Daarna volgde het dal van de desillusies. Het bleek toch veel complexer dan gedacht en de kosten wogen vaak niet op tegen de baten. Veel organisaties stopten daarom met MBT. En dat is jammer, want de ervaringen tonen ook dat MBT wel degelijk op het plateau van productiviteit kan belanden als het in de juiste situaties wordt ingezet. Welke situaties zijn dat? En waar moet je op letten als je hiermee aan de slag wilt gaan?

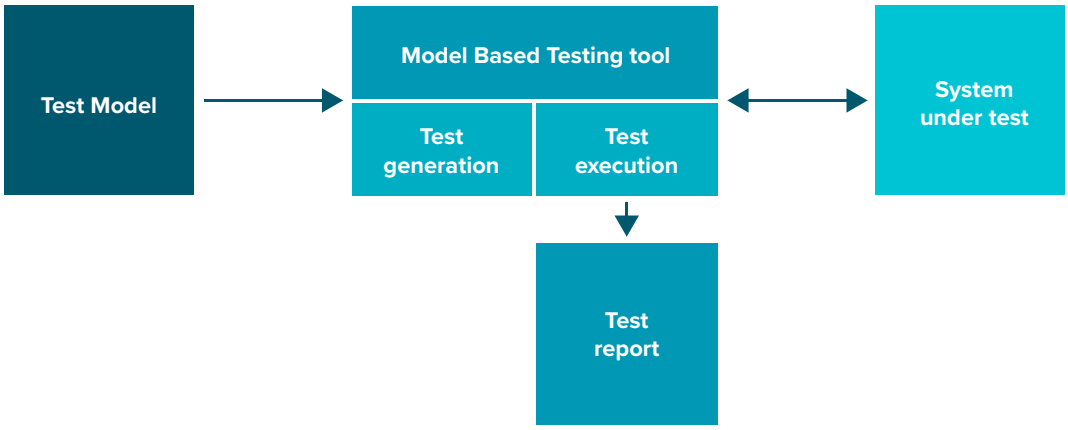
1. DE VOORDELEN VAN MODEL-BASED TESTING

MBT is een vorm van testautomatisering. Net als bij Model Driven Engineering gebruik je ook bij MBT een model als basis. Dat model dient twee doelen:

1. Het vereenvoudigt de communicatie met stakeholders omdat zo'n model veel eenvoudiger te begrijpen is dan code.
2. De tool genereert automatisch de testscripts uit het model en voert deze uit. Je hoeft je testscripts dus niet zelf te programmeren.

In het geval van MBT ontwerp je een model op basis van de voorwaarden die je aan een test stelt.
Deze voorwaarden hebben betrekking op vragen als:

- Welk onderdeel testen we precies?
- Welke risico's willen we wegnemen of beperken?



De voordelen van MBT liggen op drie vlakken:

1. Tijd- en kostenbesparing
2. Kwaliteitsverbetering
3. Verbetering van de communicatie

1.1 Tijd- en kostenbesparing

Net als bij andere vormen van testautomatisering zit de grootste winst van MBT in het automatisch uitvoeren van testgevallen, waardoor een enorme tijdsbesparing kan worden gerealiseerd. Daardoor kun je testcases sneller uitvoeren, of in dezelfde tijd meer testcases uitvoeren. In de beginjaren dacht men ook tijdwinst te realiseren door uit het model automatisch een testgeval te genereren. Dat is in sommige situaties inderdaad het geval, maar soms vergt modelleren meer tijd dan programmeren, waarover verderop meer.

Een ander groot voordeel van MBT is dat je in geval van een wijziging alleen het model hoeft aan te passen en niet het testscript zelf. Dat wordt immers automatisch gegenereerd door het model. Daardoor zijn testen veel eenvoudiger, en daardoor goedkoper, te onderhouden.

1.2 Kwaliteitsverbetering

MBT leidt op verschillende manieren tot kwaliteitsverbetering. In de eerste plaats doordat je de testdekking kunt verhogen, omdat met de automatische uitvoering meer dekking getest kan worden dan handmatig het geval is. De modellen worden soms zo complex dat die handmatig niet meer te beheersen zijn. MBT helpt je om die complexiteit te beheersen. Daardoor kun je met MBT in complexe situaties een hogere kwaliteit behalen.

Tot slot kun je al vroeg in het project testmodellen maken en zo dus ook al heel vroeg fouten in de software opsporen. Een voorbeeld is een project waarbij een communicatieprotocol moest worden getest. Bij de ontwikkeling van het testmodel bleek dat de beschrijving van het protocol onvolledig en inconsistent was. Nog voordat het testmodel af was kwam de eerste fout dus al aan het licht. Dat gebeurt door de gestructureerde manier van denken die je afdwingt met het ontwikkelen van een model.

1.3 Verbetering communicatie

Door het werken met modellen verbetert de communicatie met stakeholders, omdat iedereen naar hetzelfde model kijkt en het model voor iedereen begrijpelijk is. Neem het testen van software voor een röntgenapparaat. De domeinexperts kennen de softwarerisico's die getest moeten worden, maar ze hebben geen verstand van softwaretesten. Door samen met hen het testmodel te ontwikkelen, zien ze hoe de test precies verloopt en kunnen ze aangeven of inderdaad de juiste aspecten worden getest.

Waarom wordt MBT zo weinig gebruikt?

Als de voordelen op zoveel terreinen liggen, hoe kan het dan dat MBT zo weinig wordt gebruikt? Tenslotte is er het afgelopen decennium uitgebreid geëxperimenteerd met MBT, maar op weinig plaatsen is het echt succesvol toegepast. Er zijn verschillende redenen aan te wijzen waarom het niet lukt. De belangrijkste redenen zijn:

- Testteams, DevOps teams of projectmanagers stellen te veel doelen met MBT.
- Modellen worden niet op het juiste abstractieniveau toegepast.
- MBT wordt toegepast op gebieden waarop weinig winst is te halen.
- Er is onvoldoende volwassen tooling.

In de volgende hoofdstukken gaan we dieper in op deze oorzaken en bespreken we hoe je de succeskans kunt verhogen.

2. BEREIK VOORAF OVEREENSTEMMING OVER HET DOEL

Zoals in hoofdstuk 1 beschreven biedt MBT verschillende voordelen. Daardoor hebben diverse stakeholders vaak andere verwachtingen.

- De tester verwacht een betere kwaliteit te krijgen door het inzetten van MBT.
- De projectmanager verwacht dat het testen korter gaat duren.
- De testanalist wil de modellen vooral gebruiken als communicatiemiddel met de gebruikers of businessanalisten om zo onduidelijkheden te voorkomen

In de praktijk staat het ene doel vaak het andere doel in de weg. Als je bijvoorbeeld betere kwaliteit wilt bereiken, zul je de modellen gedetailleerder moeten maken dan de bestaande testcases. Het maken van de modellen kost tijd, waardoor de totale testduur dus niet korter wordt; in ieder geval niet in de eerste iteratie. In dat geval zal de projectmanager teleurgesteld afhaken, want hij dacht wel tijdwinst te bereiken. Focus je daarentegen enkel en alleen op tijdwinst, dan betrek je stakeholders wellicht te weinig en houd je de testmodellen te simpel, waardoor de kwaliteit onder druk komt te staan.

Het is belangrijk vooraf overeenstemming te bereiken over het doel waarvoor MBT wordt ingezet. Ligt de focus op het verhogen van de dekking, het verbeteren van de efficiency of op eerder in het ontwerptraject starten met testen? Of wordt MBT primair ingezet omdat de testcases handmatig niet meer te beheren zouden zijn? Als het doel van tevoren goed gekozen is en hier overeenstemming over is in het team, verhoogt dat de kans op succes aanzienlijk.

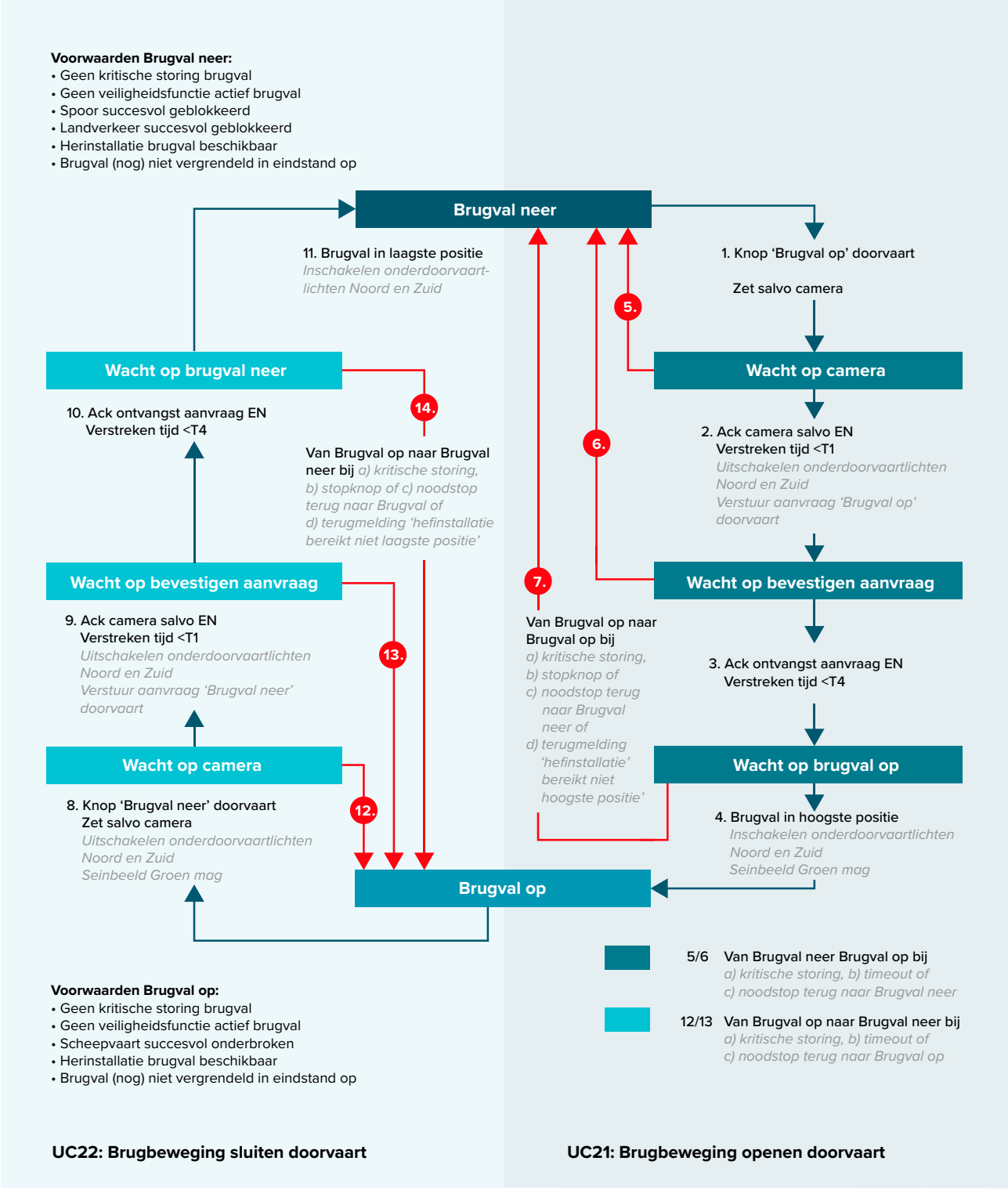
Een goed voorbeeld is een project waarbij tijdwinst het hoofddoel was. Er was al een model aanwezig vanuit de financiële regelgeving dat grotendeels hergebruikt kon worden. Eerder werden de regels van het model met de hand doorgerekend. Door dit model in te voeren in een MBT-tool, konden snel efficiëntievoordelen worden behaald.

In een ander project is MBT gebruikt met als doel: het verhogen van de kwaliteit. Er werd een model gemaakt om twee parallelle applicaties na te bootsen. Met de MBT-tool kon nu timing gesimuleerd worden, waardoor deadlock en dergelijke opgespoord konden worden. Met handmatig testen zouden deze fouten nooit aan het licht komen.

3. MAAK MODELLEN OP HET JUISTE ABSTRACTIENIVEAU

Veel mensen vinden modelleren moeilijk. Dit komt onder andere doordat het veel kennis en ervaring vereist om modellen op het juiste abstractieniveau te creëren. Helaas zijn slechts weinig testers opgeleid om goede modellen te maken. De modellen die gemaakt worden zijn prima bruikbaar voor het handmatig afleiden van testcases (als ze al gemaakt worden). Maar de complexiteit van deze modellen – of liever gezegd: het ontbreken daarvan – maakt dat ze simpelweg niet representatief genoeg zijn voor de complexiteit van een systeem in de praktijk.

Als voorbeeld nemen we een model dat is gebruikt in een project voor het ontwikkelen van brugbesturing. Het model geeft het bedienen van een brugdek weer.

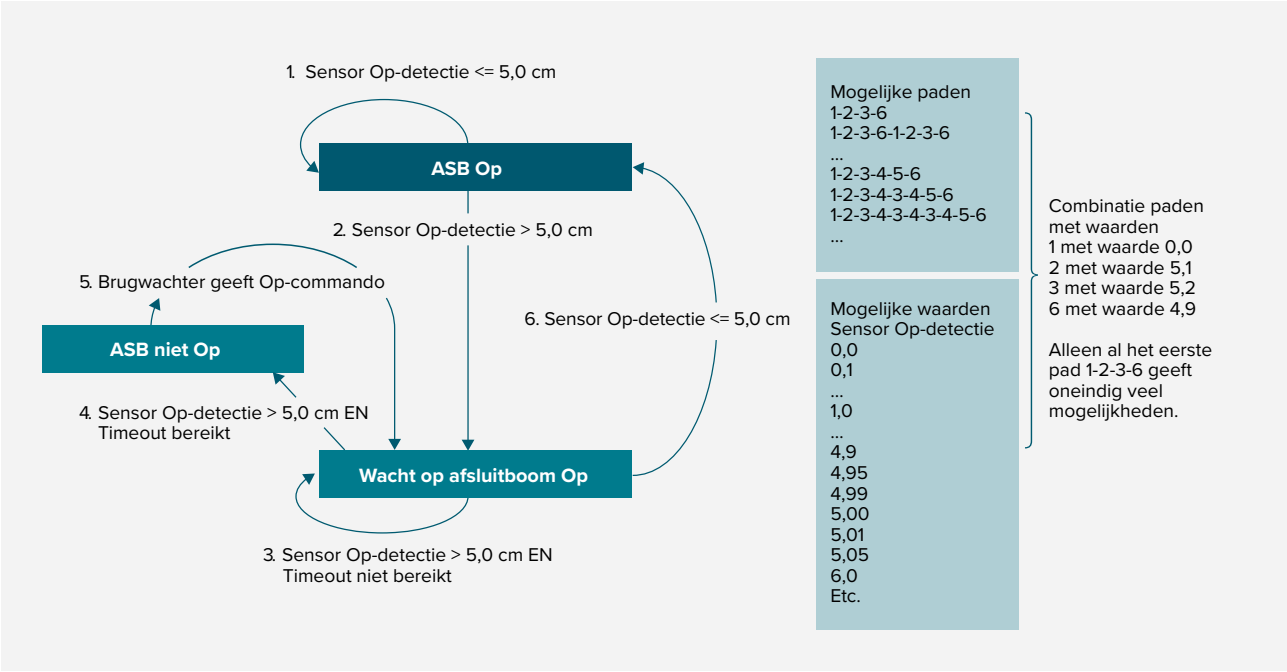


Figuur 1: Model van de besturing van een brugval.

Door de vele condities en mogelijke paden oogt dit model al best ingewikkeld, maar een echt systeem is vele malen complexer. Dit model heeft simpelweg een te hoog abstractieniveau. Je kunt het model dan verder uitwerken, maar die moeite weegt niet op tegen het handmatig genereren van testcases. Want het maken van deze complexe modellen kost in de praktijk zeer veel tijd (denk aan enkele maanden). Naast het feit dat de modellen qua structuur complex zijn, dienen in dit soort modellen ook zeer veel gedetailleerde gegevens toegevoegd te worden om ze bruikbaar te maken voor toepassing in MBT. Het maken van het model en de kosten die dat met zich meebrengt, wegen dan niet meer op tegen de baten.

Wees alert op het ontstaan van loops

In figuur 2 is een ander aspect te zien van modellen, namelijk het feit dat loops in modellen ervoor zorgen dat er theoretisch oneindig veel testcases gemaakt kunnen worden. Als dit ook nog gecombineerd wordt met oneindig veel inputdata, wordt dit wel een testcase-explosie genoemd. Je kunt daarom niet zomaar een model maken en dat draaien. Je dient er bij het instellen van de tool rekening mee te houden dat dit soort loops bestaan.



Figuur 2: Model van op-detectie van een afsluitboom.

Train testers in goed modelleren

Goed modelleren vergt dus de benodigde kennis. Door de populariteit van Model Based Engineering leren ontwerpers het vaak wel, maar voor testers is het geen basisvaardigheid. En dat is vreemd, want er worden veel modellen gebruikt in de vorm van testontwerptechnieken.

Modellen voor testen hebben andere aandachtspunten dan modellen voor ontwerpen, dus het is niet genoeg om intern in bijvoorbeeld een DevOps-team kennis over te dragen. Daardoor zijn er weinig testers die de vaardigheid echt onder de knie hebben.

Geef het model een duidelijke focus

Belangrijk bij het opstellen van modellen is dat het model een duidelijke focus heeft. Als dit ontbreekt worden vaak veel details in het model toegevoegd, wat ervoor zorgt dat het model te complex wordt. In dat geval kun je beter meerdere modellen maken met elk zijn eigen focus, zodat elk model leesbaar en bruikbaar blijft.

Houd rekening met modelleerkennis van testers

Houd daarnaast bij het bepalen van het abstractie¬niveau van het model rekening met de beschikbare kennis en ervaring van de testers met modelleren. Zoals eerder aangegeven worden de modellen vaak op een te hoog of te laag abstractieniveau gemaakt. Is het abstractieniveau te hoog, dan heeft de test geen toegevoegde waarde en zullen risico's in de software onopgemerkt blijven. En is het te laag, dan kost het te veel tijd – en dus geld – om het model te maken, waardoor de businesscase onderuit wordt gehaald.

4. KIES HET JUISTE TOEPASSINGSGEBIED

MBT leent zich lang niet voor alle situaties. Zoals eerder aangegeven zullen in veel situaties de kosten om modellen te ontwikkelen simpelweg te hoog zijn en nooit terugverdiend worden. Daarom is het belangrijk om in een project die onderdelen eruit te pikken waar MBT wél waarde toevoegt. Dat is het geval als er sprake is van hoge risico's als de software in productie wordt genomen en wanneer specifieke aspecten zoals de betrouwbaarheid en veiligheid van een systeem in het geding zijn.

Hoge productrisico's

Als een risico hoog is - vanwege een grote faalkans, hoge kosten bij falen, of beide - kun je ervoor kiezen MBT in te zetten. In dat geval wegen de kosten voor het maken van de modellen snel op tegen de baten. Is er echter sprake van een laag risico, dan is dit vaak niet het geval. Stel daarom niet als doel om MBT op je hele project toe te passen, maar vooral op die gebieden waar een verdieping nodig is in verband met het hoge risico.

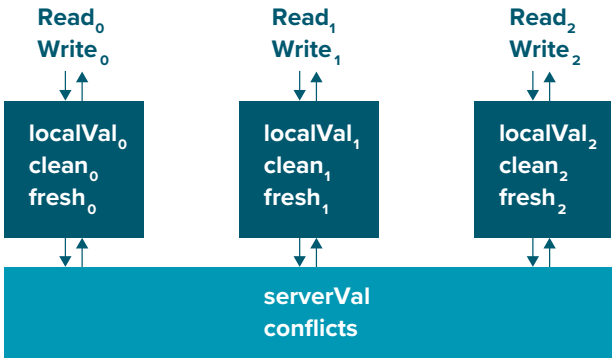
Testen van non-functionals

Uit onderzoek in 2019 door Anne Kramer, Bruno Legeard (Model-Based Testing User Survey: Results), blijkt dat 93 procent van de gebruikers MBT toepast voor functionele testen en dat slechts 21 procent MBT gebruikt voor het testen van de non-functionals, zoals performance, parallelliteit en timing. Toch lenen juist die non-functionals zich goed voor MBT.

Zo bieden sommige MBT-tools *randomisatie* aan, waardoor het mogelijk is vele paden te testen die met de hand niet of zeer lastig getest kunnen worden. Ook *parallelliteit* kan heel goed met MBT worden getest. In dat geval wordt de tool gebruikt om verschillende paden langs te gaan over de verschillende processen om hiermee te testen of er sprake kan zijn van vastlopen (deadlock). Ook in het geval van *timing issues* biedt MBT meerwaarde. Timing issues zijn handmatig lastig te testen, maar met de juiste MBT-tool kunnen verschillende timings gebruikt worden om te zien of dit een probleem geeft. In dit geval wordt de randomisatie dus gebruikt voor de timing.

Praktijkvoorbeeld testen parallelliteit en timing

In 'Model-Based Testing with TorXakis – The Mysteries of Dropbox Revisited' beschrijven Jan Tretmans en Piërre van de Laar hoe je parallelliteit en timing kunt testen. Figuur 3 toont het grafische model dat zij gebruiken. Het model toont de wijze waarop DropBox werkt als er vanaf meerdere computers bestanden gelezen en geschreven worden. Zij hebben een specifieke MBT-tool gebruikt dat zelf in staat is om via randomisering verschillende paden uit te voeren over meerdere processen.



Figuur 3: Model voor Dropbox.

5. TOOLING

Een belangrijke reden dat MBT niet goed wordt toegepast is het feit dat de tooling nog niet volwassen genoeg is. Er is veel tooling beschikbaar voor gebruik met verschillende modelleer¬technieken, alleen is die nog steeds erg beperkt. Enkele zaken waar tools nog niet goed genoeg mee omgaan, zijn de volgende:

- Sommige tools bieden alleen een tekstinterface. Voor het onderhouden van een model is het vrijwel onmisbaar om ook een grafische wijze van modelleren beschikbaar te hebben, zeker in het geval van realistische c.q. complexe modellen.
- Om iets complexere modellen te kunnen gebruiken, is het nodig dat de tools ook een manier hebben om daarmee om te gaan. Een manier om dat te doen is het gebruik van hiërarchieën van modellen. Slechts enkele tools ondersteunen dit.
- Het varieert enorm hoezeer de tools het testen automatiseren. Sommige genereren alleen testgevallen, terwijl andere direct aangesloten kunnen worden op de System Under Test (SUT). Andere tools kunnen aangesloten worden op een SUT, maar alleen via een andere tool. Zoals eerder aangegeven is de echte winst te halen uit het automatisch uitvoeren van de testgevallen. De tool dient dat dus te ondersteunen.
- De rapportagemogelijkheden van de tools blijven vaak beperkt tot het aangeven dat een testgeval niet goed doorlopen is. In andere gevallen komt er een getalletje uit dat aangeeft wat de coverage is (bijv. path coverage). Naast het feit dat dit veel werk geeft om uit te zoeken waar het probleem zit (in het SUT of in het model), is het ook geen rapportage die gebruikt kan worden voor de aansturing van de testen.
- Tools bieden maar één modelleertechniek aan en niet meerdere. Aangezien je bij het beperken van product-risico's vaak meerdere technieken gebruikt of zelfs technieken combineert (data met gedrag), kun je in dat geval niet volstaan met één techniek. Juist in deze gevallen is de verwachting dat tools kunnen helpen.

De afgelopen jaren zijn er nieuwe tools gekomen, maar die lossen de eerdergenoemde problemen (nog) niet op. Er is tot op heden niet één tool die alle gewenste functionaliteiten in zich heeft en die dus in vrijwel alle gevallen ingezet zou kunnen worden. Wel is het zo dat de ene tool beter past bij het ene project en de andere tool bij een ander project. Op die manier kun je op een constructieve manier omgaan met de beperkingen die de tools kennen. Het helpt in ieder geval als je de verschillende tools en hun sterke en zwakke kanten kent, zodat je een goed afgewogen besluit neemt van welke tool je inzet.

6. RANDVOORWAARDEN VOOR SUCCES

MBT biedt mooie kansen om in specifieke situaties de kosten van testen te verlagen en/of de kwaliteit ervan te verhogen. Omdat de methode in de praktijk vaak in de verkeerde situaties is ingezet, zijn veel organisaties voortijdig met MBT gestopt. En dat is jammer, want met name in situaties waar de risico's hoog zijn en waar specifieke non-functionals moeten worden getest, biedt MBT veel meerwaarde en zelfs mogelijkheden om cases te testen die op andere manieren helemaal niet te testen zijn.

Adviezen

Wil je aan de slag met MBT, vul dan in ieder geval de volgende randvoorwaarden goed in:

- Kies een duidelijk **gefocust doel**, zoals het verbeteren van kwaliteit of de efficiëntie. Zorg dat iedere betrokken stakeholder zich in dat doel kan vinden. Wanneer het afgesproken doel kwaliteitsverbetering is, maar een projectleider door druk van zijn opdrachtgever toch te veel gaat focussen op tijdwinst, zal de kwaliteit tegenvallen.
- Zorg voor voldoende kennis over en ervaring met het maken van modellen. Zonder deze kennis is het haast onmogelijk om de **benodigde abstractie** in het model te bepalen. Huur deze kennis eventueel in.
- Hou de **scope beperkt** en richt je op specifieke delen of aspecten van het systeem. Denk hier aan delen met een hoog productrisico of non-functionals.
- Zorg dat **toolkennis** beschikbaar is, want zoals beschreven hebben alle tools beperkingen. Huur de kennis eventueel in.

Waarom ICT?

Het automatiseren van een complex proces tot een werkend systeem is één. Maar hoe creëer je een ICT-oplossing die veel meer oplevert? Hoe zorg je voor meer snelheid? Meer gemak? Meer duurzaamheid? Meer rendement? Dát is pas een uitdaging.

ICT Group gaat die uitdaging graag aan. Want hoe ingewikkelder het project, hoe enthousiaster wij worden. En hoe ambitieuzer de doelstelling, hoe meer wij onze grenzen verleggen. Dát is wat ons drijft. En dat is waarom wij al meer dan 40 jaar succesvol zijn in technologische en industriële markten. Met meer dan 1400 professionals helpen we bedrijven, producten en projecten graag verder met slimme, innovatieve, integrale en vooral uitdagende ICT-oplossingen.

Over de auteur

Eduard Hartog: Na 16 jaar als ontwikkelaar en projectleider gewerkt te hebben, heeft Eduard uiteindelijk gekozen voor het kwaliteitsvak. Inmiddels heeft Eduard meer dan 15 jaar ervaring met zowel testen als kwaliteitsmanagement in het technische domein. Hij werkt veel in multidisciplinaire projecten met een installatietechnische component, zoals tunnels, spoor en distributiecentra. Daarnaast werkt hij in de communicatie- en offshore sector. Binnen TestNet is hij actief binnen de werkgroep Model-Based Testing.

eduard.hartog@improveqs.nl
+31 (0)6 23237330