

Automatisch testen met de testspecificatietaal TTCN3

Mark van der Zwan

Het is inmiddels een oud en bekend verhaal. Het belang van software in ons dagelijks leven groeit iedere dag. Sterker nog, de hoeveelheid software in tv's volgt vrij nauwkeurig de wet van Moore. Afgelopen november liet Hans Aerts van Philips op het Bits&Chips-managementevent zien dat de hoeveelheid software in een tv is opgelopen van 8 kbyte in 1985 tot 30 Mbyte dit jaar. De hoeveelheid fouten per 'kilo' software is tegelijkertijd al een groot aantal jaren nagenoeg constant. Dit onderstreept de noodzaak van gestructureerd en zo vroeg mogelijk testen.

De huidige hoeveelheid software in producten maakt het echter zeer lastig om met klassiek handmatig testen voldoende inzicht te krijgen in de kwaliteit. Het aantal requirements voor de huidige derde generatie mobiele telefoons is verdubbeld ten opzichte van de eerste generatie. De snelheid waarmee nieuwe producten verschijnen op deze markt vereist van leveranciers dat ze producten ontwikkelen terwijl de standaarden waaraan de telefoons moeten voldoen nog verre van stabiel zijn. Dit levert een explosieve groei van steeds veranderende eisen aan mobiele telefoons, bepaalde het European Telecommunications Standards Institute (ETSI, www.etsi.org). Deze toename in complexiteit is dusdanig dat het klassiek testen onhaalbaar is. De aanpak moet veranderen om nog serieus een uitspraak te kunnen doen over de kwaliteit van het uiteindelijke product.

De benodigde aanpassing geldt temeer omdat er veel producten voortbouwen op eerdere versies, waarbij bestaande (ingekochte) software na gedeeltelijke wijziging wordt hergebruikt. Het is daarom steeds belangrijker om na een aanpassing te testen of de ongewijzigde onderdelen nog steeds de beoogde functionaliteit en kwaliteit bieden.

Reacties ontwikkelaars en testers

Ondertussen zit de software-industrie natuurlijk niet stil. Naast de al genoemde inspanning om software te hergebruiken, richt ze haar energie aan de ontwikkelzijde steeds meer op het correct vastleggen van de vereiste software- of systeemkwaliteit en de vertaling daarvan in een gedegen ontwerp. UML-diagrammen of de resultaten van een andere formele specificatiemethode vormen de basis voor de uiteindelijke code. Iedere tester die met dergelijke specificaties heeft gewerkt droomt van een situatie waarin de eenmaal opgestelde specificaties als basistestontwerp (her)bruikbaar zijn en blijven. De testontwerpen zelf leiden semi-automatisch tot testgevallen, die met minimale handmatige interactie uitvoerbaar zijn.

Op dit moment is er nog geen tool, taal of platform dat bovenstaande volledig mogelijk maakt, al zijn er belangrijke stappen gezet. Zo is er het UML Testing Profile: een profiel waarin testarchitectuur, testdata en testgedrag zijn vast te leggen op basis van een UML-ontwerp. Het Europese TT-medal-project (www.tt-medal.org) gaat nog een stapje verder. Dit samenwerkingsverband tussen de industrie, IT-dienstverleners en wetenschap heeft tot doel het vertalen van formele (UML) specificaties naar testeisen gebruikmakend van testtechnieken. Via een testgeneratiemechanisme worden de testspecificaties vervolgens vertaald in testcode. Uitgangspunt voor deze vertaalslag is de Europese testspecificatietaal *Testing and Test Control Notation* (TTCN-3, www.ttcn-3.org). Deze relatief nieuwe testspecifieke taal, oorspronkelijk afkomstig uit de telecomindustrie en gericht op het testen van protocollen, wordt inmiddels ook binnen andere gebieden toegepast zoals breedbandtechnologie, Cobra-gebaseerde applicaties, internetprotocollen, mobiele telefonie en WLAN, . Deze toepassingen hebben allemaal gemeen dat er grote hoeveelheden interacties worden verwerkt die directe invloed hebben op ons dagelijks leven. Het falen van dergelijke systemen kan daarmee de continuïteit van de aanbieder ernstig in gevaar brengen.

Specificeren in TTCN-3

Op basis van een in TTCN-3 opgestelde testspecificatie kan door middel van een TTCN-3compiler code worden gegenereerd om de software te testen. Verschillende aanbieders hebben zowel TTCN-3-editors als -compilers gebouwd. Er zijn compilers voor onder meer C, C++, IDL, Java en XML.

Het specificeren van testgevallen in TTCN-3 is platformafhankelijk. Het is mogelijk om gebruik te maken van testspecifieke statements en constructies zoals *testverdict* en *result-based branching*. Zo kan de software testgevallen wel of niet aanroepen afhankelijk van de uitkomst (testverdict) van een voorgaand testgeval. Een TTCN-3-module bestaat hiervoor uit een definitie- en een gedragedeelte. Zo kan op eenvoudige wijze de communicatiestructuur en de inhoud van berichten worden vastgelegd, inclusief het gedrag van het te testen systeem.

Om herbruikbaarheid te garanderen blijven testgevallen generiek door het gebruik van wildcards.

Recycling is hierdoor mogelijk in andere testfasen en zelfs in andere projecten. De testgevallen krijgen pas een fysiek jasje op het moment dat ze in runtime worden gebruikt.

TTCN-3 en UML

De grafisch ingestelde Unified Modeling Language (UML, onderhouden door de Object Management Group, www.omg.org) is erop gericht om ontwikkelaars te ondersteunen in het opstellen van requirements en ontwerpen, los van de uiteindelijke code. In structurele en gedragsdiagrammen worden relaties tussen objecten en systeemgedrag vastgelegd. Bij de ontwikkeling van UML 2.0 is rekening gehouden met de TTCN-3-standaard en mede hierdoor sluiten beide talen goed op elkaar aan. Het UML Testing Profile is als notatiewijze een ander voorbeeld van de invloed die testen op de ontwikkelzijde heeft. Voor enkele van de meest gebruikte UML-diagrammen is in de tabel de relatie met bekende testtechnieken aangegeven.

UML-diagrammen	Doel van het diagram	Equivalence partitioning	Grenswaarde-analyse	Toestandovergangstests	Test use cases
Class-diagram	Identificeert entiteiten en relaties in praktijk	++	++		
Activity-diagram	Toont datastroom en/of controlestroom	+	+		+
Use case-diagram	Toont de diensten die gebruikers kunnen aanvragen				++
State-diagram	Toont de levenscyclus van een object	+	+	++	
Sequence-diagram	Toont overdracht berichten tussen objecten	+	+	+	++

Relatie tussen veel gebruikte UML-diagrammen en testtechnieken

De tabel laat zien dat een Sequence-diagram (ook wel *Message Sequence Chart* genoemd) goed aansluit op de testtechniek Test use cases. TTCN-3-editors ondersteunen dan ook de grafische notatiewijze zoals gebruikt in sequence-diagrammen om testcases te beschrijven.

Pro's en cons

De voordelen van de TTCN-testspecificatietaal zijn duidelijk. De tester is geen tijd meer kwijt met het creëren van de fysieke testscripts, het uitvoeren van de eerste falende testruns en updaten van de scripts op basis daarvan. Als er eenmaal een formele testspecificatie is, kan op basis hiervan code worden gegenereerd die de module test op het moment dat de ontwikkelaar de eisen heeft opgesteld. Dit maakt het voor de tester ook makkelijker: een wijziging in testspecificaties is zonder veel extra inspanning om te zetten in aangepaste testscripts.

TTCN-3 is niet gebaseerd op een specifieke testtechniek en ondersteunt daardoor ook niet het ontwerp van testspecificaties. Het selecteren van de juiste testtechniek voor een 'systeem onder test' zal een zaak blijven van de testexpert.

TTCN-3 komt uit de telecomindustrie, waardoor de nadruk in de voorgangers (TTCN en TTCN-2) lag op het testen van protocollen. Momenteel vindt TTCN-3 ook zijn weg naar internetprotocol-, module- en integratietesten. Zo wordt het in de automotive-industrie gebruikt om de verschillende softwaremodules in auto's in hun samenhang te testen, zonder dat alle modules al beschikbaar zijn. Bovendien maakt de standaardisatie in deze industrie het mogelijk om systemen te bouwen met ingekochte *off the shelf* modules.

Toekomst

De technische automatisering heeft inmiddels de noodzaak van standaardisatie en automatisch testen onderkend. In een vroeg stadium biedt TTCN-3 de mogelijkheid om modules te testen en later te integreren. Ook op systeemniveau is TTCN-3 toe te passen voor het testen van performance, robuustheid en het steeds opnieuw uitvoeren van regressietesten. Tegelijkertijd biedt de testspecificatietaal voldoende aanknopingspunten voor de kantoorautomatisering om ook in deze wereld te kunnen bijdragen aan de noodzaak om steeds effectiever en efficiënter te gaan testen.

Mark van der Zwan is testspecialist bij Improve Quality Services(www.improveqs.nl).

Redactie Alexander Pil

Mark van der Zwan
Improve Quality Services BV
Waalreneweg 39
5554 HA Valkenswaard
mzw@improveqs.nl