Klomps & Snels Klomps & Snels Klomps & Snels

Klop-en-probeer generatie

"En, Harry, hoe staat het met het contactpersoonscherm?" "Ja, bijna klaar. Nog even een keuzeboxje testen.... hè, wat duurt dat runnen toch lang... hij knalt eruit - ah, vergeten de check in te bouwen voordat ie de gegevens opslaat. In 5 minuutjes geregeld, opnieuw runnen duurt alleen zo lang... hè, klapt-ie er weer uit. Hmmm..."

"Zeg, Harry, kun je ook aangeven hoe lang het nog duurt voor het klaar is?" "Hmmm, tja dat komt door deze vervelende ontwikkelomgeving. Vroeger kon ik het zo zien door gewoon <F5> in te drukken. Maar dit werkt veel lastiger.". "Maar Harry, je weet zelf toch wel wat er allemaal moet gebeuren en of je dat al goed geprogrammeerd hebt? Of moet ik dat aan je ontwikkelomgeving vragen???"

Handig, 4GL-omgevingen! Vroeger gebruikte een ontwikkelaar iets als de VI-editor als hulpmiddel voor het coderen. Fantastische Unix-scripts en lappen code resulteerden, waarmee de compiler aan de gang ging. Deskchecking was geboden om te voorkomen dat bij het compileren keer op keer een nieuw regelnummer naar boven kwam waarop er 'iets' fout ging of, erger nog, hexadecimale dumps moesten worden uitgeplozen. Vroeger was alles beter...

Er werd niet persé betere code opgeleverd, en de codeersnelheid lag veel lager dan nu. Technologische ontwikkeling bracht versnelling door nieuwe talen en omgevingen. Nieuwe hulpmiddelen deden hun intrede, met directe controles op syntax én semantiek, maar ook de mogelijkheid om met een paar clicks een werkend programma én een bulk programmacode te genereren. En nu bevinden we ons in de tijd waarin uitwisselbare componenten gebruikt worden, zo van internet te plukken en met 'knip-en-plak' eenvoudig te implementeren. Programmeren kan bijna niet makkelijker, en je hoeft er haast niet meer bij na te denken. Gewoon maar F5-en, lekker trial-en-error-ren, dan kom je er achter of het al 'klaar' is. Hoezo gedegen denkwerk vooraf?

Het lijkt alsof dit geresulteerd heeft in een generatie iteratieve ontwikkelaars met een 'klop-en probeer'-mentaliteit. Hoewel... er is allang weer een kentering te zien. Ondersteund door methodes als RUP en DSDM is architectuur, in al zijn verschijningsvormen, zichtbaarder dan ooit voor de ontwikkelaars. Zij realiseren zich weer dat er meer tussen hemel en aarde is dan de F5-knop. Door de nadruk op het onder architectuur ontwikkelen, en door het toepassen van patterns en methodes die het voorgaande simpelweg afdwingen, doet 'klop-en-probeer' het niet echt meer. Bovendien blijkt bij meer-lagen architecturen ook het testen onder grotere druk te staan. Als er functionaliteit in de verkeerde laag gebouwd is, komt deze bug pas goed aan het licht in latere fases van de ontwikkeling - terwijl juist met iteratief ontwikkelen het tegenovergestelde bereikt kan worden. Tenslotte toont de 'markt' meer en meer aandacht voor kwaliteitsborging, door (on)afhankelijke certificering op een grote hoeveelheid technologische gebieden. Klop-en-probeer wordt zelfs niet meer geaccepteerd.

Door de voordelen van de 4GL-talen is een werkwijze geboren waarbij de noodzaak om eerst na te denken of de structuur van de code correct is - en dus compileert - verdwenen lijkt. De nieuwe meer-lagen ontwikkelomgevingen brengen de noodzaak tot een mentaliteitsverandering op dit vlak keihard aan het licht, maar hoe bereik je dat? Trainingen in technieken gericht op het gestructureerd ontwikkelen, zoals VSP, datamodelleren en OO, zijn het begin. Focus op echt gestructureerd programmeren en het daadwerkelijk toepassen van standaards en richtlijnen is noodzaak. Juist op al dit soort vlakken werkt een XP-achtige peer-2-peer aanpak erg goed, zolang natuurlijk wel één van de 'peers' zich bewust is van "de nobele kunst van het programmeren".

Goede ideeën, zelfs eenvoudig uitvoerbaar, maar ze laten het oude gedrag maar moeizaam verdwijnen. Na een paar jaar klop-en-probeer genot is er meer nodig om 'Nee' te kunnen zeggen tegen de <F5>toets. Moeten we ontwikkelaars dan toch weer hexadecimale dumps laten uitpluizen? Bouwen ze vast een tooltje voor...

Birgit Klomps en Tommes Snels, beiden managing consultants bij Capgemini, schrijven op persoonlijke titel over het werk in projecten, in workshops en alles wat daarbij komt kijken.