De invoering van testautomatisering

In de vorige blog hebben we gekeken naar de rol van testautomatisering binnen CI/CD.

In deze blog, de laatste van de reeks, kijken we naar **het groeipad bij de invoering van testautomatisering**.

Groeipad invoering testautomatisering

Voor groei is het goed je te realiseren dat verandering beter gaat als er pijn/noodzaak is. Dit mechanisme kun je gebruiken door, met het model en gedachtengoed van de testpiramide in het achterhoofd, tests te implementeren met oog voor beheertijd en doorlooptijd.

Als ondersteunende maatregel is het verstandig om root-cause analyse uit te voeren op elke door een geautomatiseerde test gevonden bug. Stel bij elke bug de vraag of deze met een ander type test (lager in de testpiramide) gevonden had kunnen worden. Op deze manier bouw je de business case voor de investering in de volgende verbeterstap.

Daarnaast help het om tijdens de sprint refinement expliciet stil te staan bij het opstellen van een teststrategie per sprint en per user story. Hierdoor ontstaat een grotere bewustwording voor het test(automatiserings)vraagstuk bij het team. Naarmate de bewustwording groeit en er meer ervaring is opgedaan met de geïmplementeerde geautomatiseerde tests, kunnen meer testniveaus van de piramide opgenomen worden in de teststrategie.

Afhankelijk van de context zijn er twee aanpakken voor groei.

1. ***Green field: er is nog geen testautomatisering.***

Start dan op alle lagen van testpiramide tegelijk. De slagingskans van deze aanpak hangt sterk af van zowel het team (kennis van testautomatisering en kwaliteitsbewustzijn) als het product (testbaarheid) en de kwaliteit en beschikbaarheid van testomgevingen. Als je met een nieuw team aan een nieuw product begint, heeft de integrale aanpak de voorkeur.

1. ***Als er al een begin is gemaakt met testautomatisering***

Vaak hebben teams al een begin gemaakt met testautomatisering vanuit de problematiek dat handmatig testen te veel resources (tijd en menskracht) kosten. In een agile scrum setting betreft het dan vaak de regressietest. Meestal zijn in deze situatie de handmatige tests 1:1 geautomatiseerd met een tool of framework. Vaak ontstaan met deze aanpak problemen rond beheerbaarheid en doorlooptijd, we zitten namelijk bovenaan in de testpiramide. In deze context is het “omlaag duwen” van tests in de testpiramide een goede aanpak.

In de **eerste stap** kijken we met een scherp oog naar de gebruikte tool of framework. Als een tool of framework quick en dirty is geïmplementeerd zal eerst een refactorslag uitgevoerd moeten worden om het schaalbaar, stabiel en beheerbaar te maken. Als er een tool is gebruikt (bv een record en replay tool) zal gekeken moeten worden of deze oplossing voldoende toekomstvast is. Vaak ontstaan met deze aanpak problemen rond schaalbaarheid, beheerbaarheid of stabiliteit. Indien nodig zal de bestaande automatisering omgebouwd moeten worden naar een nieuw framework.

Pas in deze stap goed op voor het bouwen van te veel testdekking! Bouw vooral de tests waarmee je de waarde van het product op user journeys aantoont. Houd een scherp oog voor doorlooptijd en de uitkomsten van de root-cause analyse op gevonden bugs. Op een gegeven moment kom je op een punt dat blijkt dat er winst te behalen is op een ander, lager testniveau (b.v. API). Dit is het moment om een niveau dieper te gaan in de testpiramide.

In **stap twee** is het belangrijk kennis op te doen van de productarchitectuur. Welke interfaces en API's heeft het product? Met deze kennis kan vervolgens testdekking op losse modules, op de integratie tussen modules en op interfaces en API's worden gecreëerd. Met het bouwen van testdekking op deze onderliggende lagen is het belangrijk om te kijken of al geautomatiseerde GUI tests overbodig worden. Een GUI edge case op een invoerveld kan in deze fase vrijwel altijd vervangen worden door een edge case op de invoermodule. Daarmee kan de GUI edge case test komen te vervallen.

In **stap drie** zakken we verder af in de testpiramide. Om dat te kunnen moeten we, in nauwe samenwerking met de developers, de productarchitectuurkennis verdiepen tot het niveau van componenten en code units. Met deze kennis kun je als team kijken welke testdekking verplaatst kan worden naar unit testdekking of component testdekking. De edge case test op een invoer module kan hierbij vrijwel altijd verplaatst worden naar een unittest op de invoer methode of invoer klasse.

Met het volgen van deze drie stappen ontstaat een beheersbaar groeipad naar steeds effectievere en efficiënte testautomatisering.

Laatste blog

Tot zover deze blogserie over het managen van testautomatisering. Testautomatisering is een groot onderwerp met veel facetten. Ik pretendeer niet volledig te zijn, maar heb geprobeerd verschillende, belangrijke facetten en hun samenhang te duiden zodat u voldoende basiskennis heeft om testautomatisering te kunnen managen.

Als u wilt, kunt u deze blogreeks downloaden als whitepaper.

Veel succes en heeft u vragen over testautomatisering dan help ik of één van mijn collega's u

graag verder!

Wim ten Tusscher