Er is mij gevraagd iets te schrijven over een typische dag uit het leven van een **Test Automation Engineer**. Dit soort dingen schrijf ik graag, maar ik wil toch vooraf een paar aantekeningen maken.

Typische dagen bestaan gelukkig niet, het werk zou erg eentonig worden als je je werkzaamheden in een typische dag kan vatten. Daarnaast, ik geloof niet dat Test Automation Engineer een aparte functie zou moeten zijn. Anno 2019 mag van testers verwacht worden dat ze minstens basale programmeerkennis hebben.

De opkomst van testautomatisering

Bij de refinement vraag je je als tester niet alleen af of er voldoende informatie is om te kunnen testen, maar ook om te automatiseren. Als er al een bestaand testframework is, kan het zijn dat nieuwe user stories impact hebben op al bestaande tests. Als tester beheer je dit framework en geef je ook aan hoeveel extra werk nieuwe user stories opleveren. Zeker bij planningsessies is het van belang rekening te houden met het feit dat testautomatisering niet altijd tijdbesparend is, maar ook extra werk op kan leveren.Dat gezegd hebbend, ik ben al ruim tien jaar als tester aan het werk. De eerste jaren alleen als handmatig tester en de afgelopen paar jaar steeds meer ook als Test Automation Engineer. Ik wil dit artikel dan ook vooral gebruiken om te beschrijven hoe de werkzaamheden van een tester door de opkomst van testautomatisering zijn veranderd. Ik spreek over traditionele testers als ik het heb over mijn rol van een jaar of vijf geleden, waarin ik vrijwel alles via de front-end handmatig testte. De moderne tester heeft automatseringskennis en past deze ook toe

Het begin van de sprint staat voor ‘traditionele’ testers in het teken van het uitschrijven van tetscases. Als ‘moderne’ tester bouw je ook de testcases al zoveel mogelijk in je framework, zodat deze direct na oplevering kunnen worden gerund. Het bouwen van testscripts, zeker voor testers met nog niet zoveel automatiseringskennis, of in een taal waar ze minder bekend in zijn, kan erg tijdrovend zijn.

Over het algemeen is testautomatisering vrij eentonig, maar de 20/80 regel is hier ook van toepassing. Een fractie van de hoeveelheid werk slokt het gros van de tijd op. Zeker toen ik net als testautomatiseerder was begonnen zat ik continu op stackoverflow naar oplossingen te zoeken.

Afstemmen met de organisatie

In een volwassen organisatie zullen regressietests onderdeel zijn van de pipeline van je build server. Onderdeel van de werkzaamheden van de moderne tester is het afstemmen met de organisatie (business en development) welke tests bij welke deployment op welke omgeving moeten worden uitgevoerd. Er is een spanningsveld tussen de tijd die het duurt om regressietests te draaien en het risico dat door de tests wordt afgedekt. Meer risico afdekken betekent een langere deployment tijd. De belangen kennen, deze tegen elkaar afwegen en hierover adviseren of beslissen; het is allemaal van belang voor de moderne tester.

Tests kunnen om wat voor reden dan ook falen. Als een deployment mislukt omdat tests niet het gewenste resultaat geven, wordt van je verwacht deze snel te analyseren. Is er door de deployment daadwerkelijk iets omgevallen? Heeft een aanpassing in de code een onvoorzien effect gehad op andere functionaliteit. En zo ja, wat is dit effect en is dit erg? Moeten tests worden aangepast, of moet de release opnieuw worden uitgevoerd?

Mogelijk lukt het niet om alles te automatiseren binnen een sprint. Schakel tijdig hulp in bij de rest van het team en/of overleg met de product owner of een deel van de testautomatisering als technical debt op de backlog kan.

Over het algemeen vindt er een review proces plaats op je testcases. Er zal af en toe afgestemd moeten worden met andere testers of ontwikkelaars als zij feedback hebben op je testcases en hier aanpassingen op willen hebben. Andersom geldt natuurlijk ook dat je zelf gevraagd wordt een review te doen op testcases van je collega’s. In mijn ervaring gaat dit altijd via GitHub.

Als er door meerdere collega’s in één framework wordt gewerkt, kan het nodig zijn dat er afspraken gemaakt worden over code standaarden. Iedere (test)automatiseerder heeft een eigen stijl, maar werk moet wel kunnen worden overgedragen aan elkaar.

Exploratory testen

Het zijn eigenlijk best veel taken die er voor een moderne tester bij zijn gekomen ten opzichte van de traditionele tester. Een valkuil waarin ik zelf wel eens ben getrapt, is dat je te

veel in code zit en daardoor de feeling met het product wat verliest. Ik kom uit een school van exploratory testing en merk dat ik de meeste bugs of verbetersuggesties toch vind door één of twee uur een specifiek onderdeel van het systeem handmatig via de front-end te testen. Ik zorg ervoor dat ik ondanks de automatiseringstaken nog altijd tijd vrij houd voor de exploratory testsessies.

Altijd blijven leren

Voor de testautomatiseerder geldt ook, misschien nog wel meer dan voor de traditionele tester, dat je altijd moet blijven doorleren. Met enige regelmaat volg ik dus in de avond nog cursussen bij Polteq of bezoek ik avonden van het testautomatiseringsgilde, waar collega's ervaringen uitwisselen met elkaar. Ook gebeurt het wel eens dat ik me 's avonds op een probleem stort waar ik overdag niet uitkwam. Af en toe doe ik ook een online cursus. Er is van alles te vinden, codeacademy vind ik zelf wel handig om met een nieuwe taal te leren of dieper op materie in te gaan.

Maar de meeste avonden breng ik gelukkig nog altijd door met een boek, sportend, Netflix kijkend of een game.

Stefan Croes