Afwegingen bij het testen van een software applicatie op een context-driven manier

Afwegingen bij het testen van een software applicatie op een context-driven manier

Auteurs:

- Ruud Cox - Ruud.Cox@improveqs.nl
- Joris Meerts - Joris.Meerts@improveqs.nl
- Huib Schoots - Huib.Schoots@improveqs.nl

Introductie

Binnen het kader van context-driven testen (Kaner, Bach, & Pettichord, 2001) wordt de werkwijze die men toepast bij het testen van software bepaald door de situatie waarin de tester verkeert. Een kwalitatief goede aanpak wordt niet gestuurd door een voorgeschreven proces of door een verzameling van handelingen die men gewoon is te doen. Een goede benadering ontstaat juist door het inzetten van vaardigheden die ervoor zorgen dat het testen aansluit bij de omstandigheden van het softwareproject. Binnen het kader van context-driven testen wordt gesproken over een breed scala aan vaardigheden, waaronder kritisch denken, modelleren en visualiseren, het maken van notities en het toepassen van heuristieken. Het is niet eenvoudig om deze vaardigheden te leren. Een manier om ze aan te scherpen is het maken van oefeningen en het bespreken van de resultaten. Dit was het vertrekpunt voor een bijeenkomst van vier testers bij Improve Quality Services. In het onderstaande verslag wordt dieper ingegaan op de oefeningen die tijdens die bijeenkomst gedaan zijn en op de resultaten daarvan.

Doel van dit verslag

Zoals we in de introductie al aangaven is het niet eenvoudig om de vaardigheden die bij context-driven testen horen te leren. Door beoefenaars van context-driven testen wordt regelmatig verwezen naar vakliteratuur die deze vaardigheden beschrijft. Maar het toepassen ervan gaat met vallen en opstaan. Daarom is het belangrijk de praktijk te simuleren en te leren van de oefeningen die we doen. Dat is het doel van de georganiseerde bijeenkomst. In dit verslag delen we de stappen die we hebben gevolgd en de resultaten van die stappen, bijvoorbeeld in de vorm van notities, schetsen of modellen. We delen onze ervaringen om het toepassen van de vaardigheden in de praktijk eenvoudiger te maken.

De bijeenkomst

De bijeenkomst is gehouden op 14 februari 2017 op het kantoor van Improve Quality Services in Eindhoven. De deelnemende testers, Jos Duisings, Ruud Cox, Joris Meerts en Huib Schoots zijn allen medewerkers van Improve Quality Services.

Achtergrondinformatie

Datum	14 februari 2017
Locatie	Improve Quality Services Eindhoven
Aanvang	09:00 uur
Einde	17.00 uur
Team A	Jos Duisings, Ruud Cox
Team B	Joris Meerts, Huib Schoots

De opdracht

De opdracht van de bijeenkomst is het selecteren en uitvoeren van enkele testen op een software applicatie. De opdracht wordt uitgevoerd door twee teams van elk twee testers. Dit maakt het

mogelijk om verschillende benaderingen uit te voeren. Bovendien maakt dit het mogelijk om tussen de teams onderling feedback te geven.

Opdeling in sessies

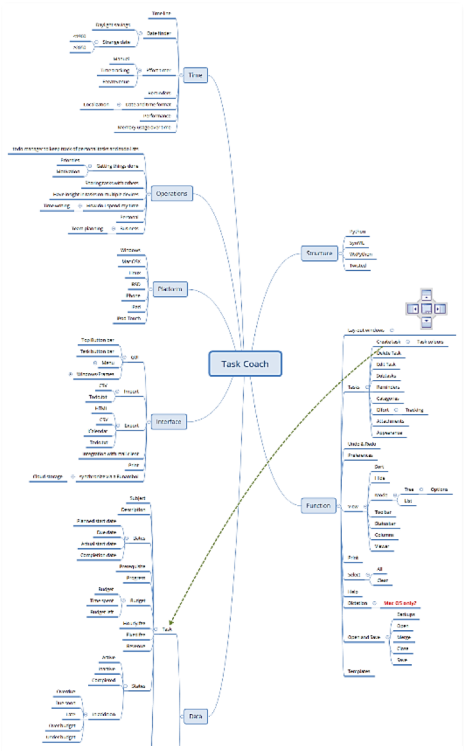
De bijeenkomst is vooraf opgedeeld in sessies. De sessies hebben elk een eigen doel en hun eigen evaluatie.

- Welkom & introductie
- Maken coverage outline (per team)
- Debriefing coverage outline
- Opstellen test strategie (als groep)
- Debriefing test strategie
- Selecteren van enkele charters
- Uitvoeren van een test sessie (charter) per team.
- Debriefing test sessies
- Retrospective

Introductie van de te testen applicatie

Voorafgaand aan de bijeenkomst is de te testen applicatie geselecteerd. Het gaat om de applicatie Task Coach (Task Coach, 2018), een open source beheerder van to-do-lijstjes. Deze software is openbaar beschikbaar en draait op meerdere platformen (Windows en Mac OS). De applicatie is relatief eenvoudig maar biedt toch voldoende complexiteit om diepgaand te kunnen testen. Volgens de omschrijving op de website is Task Coach “een eenvoudige open source to-do-manager die persoonlijke taken en to-do-lijstjes bijhoudt. Het is ontworpen voor samengestelde taken en biedt ook nog de mogelijkheid om gespendeerde tijd bij te houden en categorieën en notities aan te maken.”

De deelnemers hebben niet eerder met Task Coach gewerkt en zijn dus onbekend met deze specifieke software.



Figuur 1 Mind map van de applicatie Task Coach

Het maken van een productschets

Omdat Task Coach voor alle deelnemers nieuw is, zal de software verkend moeten worden. Daarna pas kunnen we meer zeggen over wat er in de gegeven tijd getest kan worden. Zo'n verkenning van het product kan op veel verschillende manieren. Tijdens de bijeenkomst hebben we gekozen voor het maken van een 'product coverage outline', ingegeven door het Heuristic Test Strategy Model van James Bach (Bach, 1996). De product coverage outline is een overkoepelende productschets waarbij de functionaliteit van de software opgedeeld wordt in zeven verschillende categorieën. De categorieën zijn beschreven in het ezelsbruggetje SFDIPOT. De software tester wordt door dit ezelsbruggetje eraan herinnerd dat hij bij het in kaart brengen van een product kan kijken naar Structure, Function, Data, Interfaces, Platform, Operations en Time. Door de applicatie te bekijken vanuit deze perspectieven ontstaat een relatief complete schets van wat het software product allemaal kan. Het ezelsbruggetje dient echter niet alleen ter verkenning. De 'landkaart' van de applicatie die op deze manier tot stand komt kan ook gebruikt worden om de dekkingsgraad van de testen te visualiseren en voor het selecteren van de uit te voeren testen.

De oefening

In overleg is besloten dat beide teams een half uur de tijd krijgen voor het opstellen van een productschets en dat elke team de vrijheid heeft wat betreft de manier waarop de schets wordt uitgewerkt. De keuze voor een tijdslimiet van een half uur is hoofdzakelijk ingegeven door het feit dat aan het einde van de dag er ook software getest moet zijn. Er is geen ruimte om erg veel meer tijd te steken in het maken van de productschets.

Een half uur blijkt te weinig tijd voor het in kaart brengen van een applicatie die toch behoorlijk veel functies bevat. Team B heeft ervoor gekozen om een mind map te maken waarin de zeven categorieën worden uitgewerkt. Deze mind map is na een half uur bij lange na niet af. Vooral de tak 'Functies'—waarin de functies van de applicatie zijn beschreven—is erg groot en heeft een diep geneste structuur. Om deze structuur uit te werken klikt het team door de applicatie heen en legt het de functionaliteit (in tekst of in afbeelding) vast in de mind map.

Vanwege tijdgebrek presenteert team B een productschets die niet af is. Hierdoor ontstaat een discussie over de eisen die aan de mind map gesteld mogen worden gegeven de beschikbare tijd. De eisen, zoals bijvoorbeeld volledigheid, zijn te relateren aan het doel van de mind map. Door een discussie wordt duidelijk dat er geen helder doel voor het opstellen van de productschets is gedefinieerd en dat hierdoor het resultaat van de opdracht moeilijk te beoordelen is.

Voorkeur voor de mind map

Een mind map is een gereedschap dat met enige regelmaat door software testers gebruikt wordt voor het maken van een productschets. De boomstructuur van de mind map leent zich voor classificeren (groeperen) van eigenschappen. Het is eenvoudig om te beginnen met een lege mind map, daarin de categorieën uit SFDIPOT te plaatsen en deze verder uit te werken. Bovendien

ontstaat op deze manier een navigeerbare structuur. De mind map vormt een soort landkaart waarin naar keuze ingezoomd en uitgezoomd kan worden om de plek van iets in het geheel te bepalen.

Figuur 2 Het helpbestand van Task Coach biedt een mooi overzicht van de functionaliteit

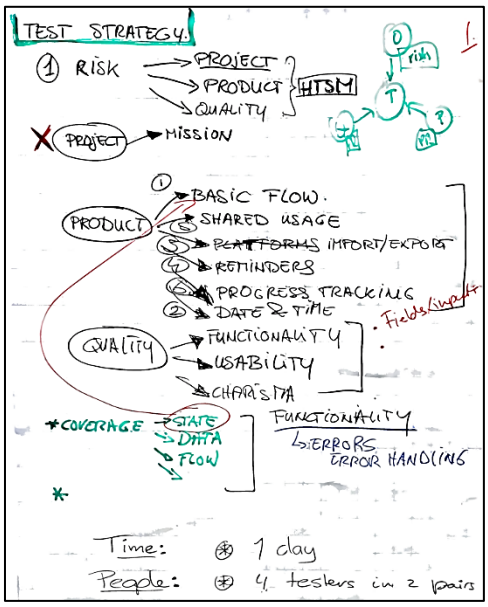
Software bestaat in essentie uit nullen en enen en het softwareproduct is dus een abstract begrip. Door het maken van de 'landkaart' krijgt de abstracte software een concrete vorm. Een mind map is verder te gebruiken als instrument voor het rapporteren over de resultaten en de voortgang van de testen. Er is dus kortom een aantal goede redenen om vanaf het begin met een mind map te gaan werken.

Team A start de sessie met het maken van een mind map maar stapt na korte tijd over naar een andere vorm. Het team komt er bij het bestuderen van Task Coach achter dat er een helpbestand aanwezig is in de applicatie. Na een korte bestudering blijkt het helpbestand een groot deel van de applicatie te beschrijven. De categorieën uit SFDIPOT komen allemaal in voldoende mate terug in het bestand en dus kiest team A ervoor om dit bestand, omgezet naar Word formaat, in te dienen als productschets. Door de categorieën in het document met verschillende kleuren te markeren wordt structuur aangebracht in het document. Bovendien biedt de inhoudsopgave een globaal overzicht van de functionaliteit in de applicatie; de details staan vermeld in de paragrafen. Zo levert team A binnen de gestelde tijd een productschets die relatief volledig is.

Het opstellen van de test strategie

Het resultaat van de eerste sessie is een productschets. Met deze productschets en met de kennis die is opgedaan tijdens de verkenning van de software is het makkelijker om te discussiëren over een test strategie. Omdat het uitputtend testen van een applicatie voor het merendeel van de software applicaties niet haalbaar is en omdat uitputtend testen in veel gevallen niet betere informatie oplevert, is het wenselijk om keuzes te maken voor wat betreft het uit te voeren testwerk. Deze keuzes vinden we terug in de test strategie.

Door het maken van de productschets hebben we inzicht gekregen in een aantal aspecten van de applicatie. Deze aspecten nemen we mee en we hopen per categorie aan te kunnen geven of deze categorie getest moet worden en zo ja met welke diepgang. De meest bepalende factor in die beslissing is het risico dat het bedrijf zou kunnen lopen bij het in gebruik nemen van de software. Risico is een veelzijdig begrip en kan alleen bepaald worden als het software product vanuit verschillende kanten wordt belicht. In ieder geval is het perspectief van de tester alleen niet voldoende om een goed beeld te krijgen van de risico's van een software product. Tijdens de tweede sessie wordt duidelijk dat een vertegenwoordiger ontbreekt die vanuit het perspectief van een gebruiker of vanuit de organisatie het risico kan beredeneren. Ook in de nabespreking van de tweede sessie komt dit punt ter sprake en concluderen we dat om die reden is de risicoschatting onvolledig is.



Figuur 3 Resultaat van de risico inschatting

Het inschatten van risico

In het Heuristic Test Strategy Model wordt gesproken over drie dimensies die een test strategie beïnvloeden. Deze dimensies zijn het project, het product en de eisen die aan dat product gesteld worden; de kwaliteitseisen. Binnen elk van deze dimensies speelt risico een rol. In de oefening besluiten we niet te kijken naar de projectrisico's. Wat betreft de productrisico's maken we een eigen inschatting met betrekking tot de categorieën van het product die het meest gebruikt worden, het meest gevoelig zijn voor fouten en waarop een eventuele fout het meeste invloed heeft. Omdat er geen gebruikers betrokken zijn bij de oefening proberen we onszelf te verplaatsen in de rol van de gebruikers. Zodoende komen de onderstaande productcategorieën aan bod.

- Het primaire proces uit de categorie **Handelingen (Operations)**. Hiermee wordt de meest gebruikte functionaliteit doorlopen.
- Het gebruiken van de applicatie door meerdere gebruikers uit de categorie **Handelingen**. We zien hier risico's op het gebied van synchronisatie die ervoor kunnen zorgen dat taken niet goed worden bijgewerkt.
- Het importeren en exporteren van taken uit de categorie **Koppelingen (Interfaces)**.
- Het omgaan met herinneringen uit de categorie **Functionaliteit (Functionality)**. Herinneringen zijn van cruciaal belang voor het niet missen van afspraken.
- Het omgaan met datum en tijd uit de categorie **Tijd (Time)**. Datum en tijd spelen een belangrijke rol in het plannen van taken, ze raken de kern van de applicatie.

Nadat we de categorieën van het product hebben benoemd, volgt een inschatting van de belangrijkste eisen die aan het product gesteld zouden kunnen worden. Ook hier is ervoor gekozen om een eigen inschatting te maken. De volgende kwaliteitsaspecten worden benoemd:

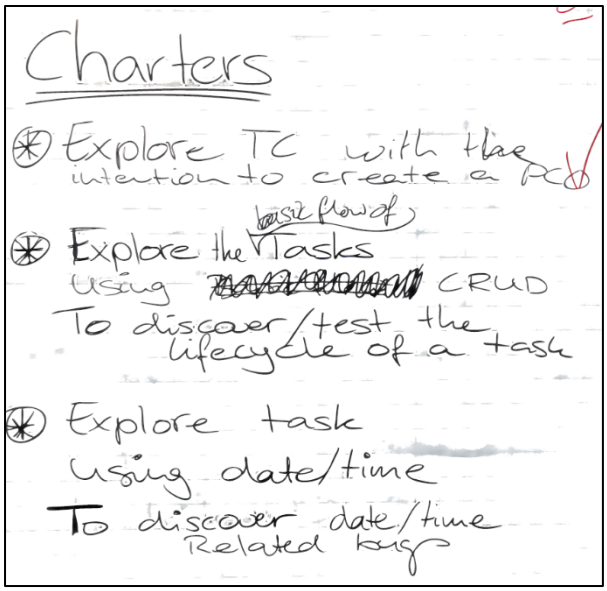
- functionaliteit,
- bruikbaarheid en
- charisma.

Dekking

Er is kort gesproken over de dekkingsgraad van de testen die we uit willen voeren. Aangezien de taken die in Task Coach onderhouden worden verschillende statussen kunnen hebben kan de tester vanuit het perspectief van statusovergangen een beeld vormen van de dekkingsgraad van de uitgevoerde testen. De statusovergangen worden geraakt in de verschillende paden door de applicatie die een gebruiker bewandelt. Deze paden helpen bij het in kaart brengen van de dekking. Verder kan er gekeken worden naar de dekking vanuit het perspectief van de testdata die gebruikt wordt.

Het opstellen van de charters

We besluiten om twee charters te maken en deze uit te voeren. We prioriteren de in de risicoanalyse genoemde productcategorieën op basis van onze eigen inzichten. Daaruit concluderen we dat het primaire proces en het omgaan met datum en tijd de twee belangrijkste productcategorieën zijn.



Figuur 4 De opgestelde charters

Deze twee categorieën vertalen we naar charters. In één charter gaan we kijken naar het primaire proces, in de andere charter wordt het omgaan met datum en tijd onderzocht. Elk team kiest een charter. Nadat de charter is uitgevoerd koppelt ieder team het gedane werk terug aan het andere team in een debrief sessie. Tijdens deze korte evaluatie doet de tester zijn verhaal en beantwoordt hij eventuele vragen. De evaluatie heeft meerdere doelen, zoals het beoordelen of de missie geslaagd is, het omzetten van nieuwe inzichten in vervolgsessies, het bekijken van aantekeningen en beschrijvingen van bevindingen en coachen van de tester. Door de debrief groeit het inzicht in de te testen applicatie.

De charter voor het testen van het primaire proces

Om een uitgangspunt te formuleren voor deze charter wordt gekeken naar de risico's die kunnen horen bij het uitvoeren van het primaire proces. Zo bestaat er een risico dat er geen taak aangemaakt kan worden. Er kunnen bijvoorbeeld ook fouten optreden bij het opslaan of wijzigen van een taak of bij het veranderen van de status van een taak. Deze overwegingen leiden tot de volgende missie die als uitgangspunt dient voor de charter:

Verken het primaire proces van de applicatie met als doel het verkrijgen van inzicht in de levenscyclus van een taak

De charter voor het testen van het omgaan met datum en tijd

Voorafgaand aan het opstellen van de charter voor het testen van het omgaan met datum en tijd worden de volgende test ideeën genoemd.

- Start- en eindtijd zijn gelijk
- De eindtijd ligt voor de starttijd
- De datum van een taak ligt ver in het verleden of ver in de toekomst
- Systeemtijd
- Werken in verschillende tijdzones
- Omgaan met wintertijd en zomertijd
- Verschillende datum formaten

Met deze test ideeën in het achterhoofd wordt de volgende charter opgesteld:

Ontdek fouten die gerelateerd zijn aan datum en tijd

Uitvoering van de charters

Het testen van het primaire proces

De charter voor het testen van het primaire proces begint met het aanmaken van een taak. Middels de button op de taakbalk worden meerdere taken aangemaakt. Vervolgens worden voor een enkele taak meerdere onderliggende taken aangemaakt, tot 8 niveaus diep. Onder één van de

onderliggende taken wordt een nieuw boomstructuur van onderliggende taken aangemaakt. Tijdens het aanmaken van de taakstructuur ontstaan vragen over maximale diepte van de taakstructuur en over de sortering van de onderliggende taken. Bovendien wordt geconstateerd dat het mogelijk is om onderliggende taken te verwijderen en vervolgens die verwijdering ongedaan te maken. Er wordt voorgesteld om deze functionaliteit in een aparte charter verder uit te werken, aangezien het vermoeden bestaat dat dit niet altijd goed gaat. De ontstane boomstructuur wordt verwijderd middels de 'Verwijderen' knop op de taakbalk. Hierna zijn alle taken verdwenen.

Op de taakbalk zit ook een knop die de mogelijkheid biedt om nieuwe taken vanuit een template te creëren. Er zijn twee default templates aanwezig. Met deze templates worden taken aangemaakt. Het blijkt dat het niet mogelijk is om onderliggende taken aan te maken vanuit een template. De vraag is waarom deze functionaliteit niet beschikbaar is voor onderliggende taken. Tot slot worden de aangemaakte taken verwijderd.

Om een beter beeld te krijgen van de werking van templates in de applicatie wordt de helptekst over templates gelezen. Bovendien wordt gezocht naar functionaliteit rondom templates in de menu structuur. Vanuit het 'Bestand' menu is het mogelijk om taken als templates op te slaan, om templates te importeren en om templates te bewerken. Er wordt een template aangemaakt.

Uit de dialoog die verschijnt als wordt gekozen voor het "importeren van", blijkt dat template bestanden de extensie '.tsktmpl' zouden hebben. Maar als een taak als template wordt opgeslagen is niet te achterhalen of van deze taak een template bestand wordt gemaakt en zo ja, waar dit bestand wordt opgeslagen. De nieuw aangemaakte templates zijn wel zichtbaar als in de werkbalk wordt gekozen voor het aanmaken van een nieuwe taak op basis van een template.

Er wordt een taak aangemaakt, opgeslagen en gekeken of deze taak geïmporteerd kan worden.

Opnieuw worden alle aangemaakte taken verwijderd door het selecteren van de menu optie Bewerken > Verwijderen. Er wordt iets verder gekeken naar de opties voor het verwijderen van taken. Het blijkt dat er een sneltoets combinatie is voor het verwijderen van taken. De combinatie is Ctrl+DEL. Het team vraagt zich af waarom er niet gewoon gebruik gemaakt kan worden van de Delete toets.

Samenvattend is er in deze sessie gekeken naar het aanmaken, bekijken, bewerken en verwijderen van taken en onderliggende taken. Er is een bevinding gedaan met betrekking tot het ongedaan maken van wijzigingen. Het blijkt namelijk dat het ongedaan maken van het verwijderen van taken in een boomstructuur van taken en onderliggende taken niet dezelfde structuur oplevert als de oorspronkelijke structuur. Naar aanleiding van de sessie wordt voorgesteld om in nieuwe charters nog uitgebreider te kijken naar het gebruik van templates. Ook de functionaliteit voor het aanmaken, bekijken, bewerken en verwijderen van taken en onderliggende taken verdient meer aandacht, met name omdat deze functionaliteit vanuit veel verschillende plekken in de applicatie aan te roepen is. Deze mogelijkheden zijn niet allemaal getest in de eerste sessie. Verder zou er een charter gemaakt kunnen worden voor het aanmaken van een taak afhankelijk van een andere taak.

Terugkoppeling

Na afloop van de sessie is het verslag van de sessie uitgebracht aan een lid van het andere team met als doel het verkrijgen van terugkoppeling over het verloop van de sessie. Uit de terugkoppeling blijkt dat er niet genoeg tijd was om de charter af te ronden. Om de charter af te ronden is nog dertig tot zestig minuten nodig. Er wordt opgemerkt dat de beschrijving van de gevonden bug niet helder genoeg is. Ook wordt duidelijk dat niet alle statussen die een taak kan hebben voorbij gekomen zijn

in de sessie. Tot slot wordt een nieuwe charter gesuggereerd voor het testen van filtering en sortering.

Het testen van het omgaan met datum en tijd

De sessie wordt gestart met een schone installatie van de applicatie. Als eerste wordt een nieuwe taak aangemaakt. De aandacht gaat uit naar het tabblad Data waarop verschillende data kunnen worden ingevoerd. In aparte invoervelden kunnen de datum en de tijd gewijzigd worden. De tijd kan gewijzigd worden middels een dropdown of door gebruik te maken van de pijltjestoetsen op het toetsenbord. Het blijkt dat '00:00' niet kan worden opgevoerd als tijd. Het blijkt dat onder het menu 'Voorkeuren' het bereik van de tijd kan worden ingesteld. Na een aanpassing kan '00:00' wel worden opgevoerd.

Er wordt geconstateerd dat de geplande startdatum en de geplande einddatum niet worden meegenomen in de berekening. Maar het is onduidelijk waarvoor deze gegevens dan wel gebruikt worden. Als de geplande einddatum in het verleden ligt dan kleurt de taak rood. Als de geplande einddatum in de toekomst ligt dan kleurt de taak paars. Opvallend is dat de geplande startdatum na de geplande einddatum kan liggen. Blijkbaar is er geen validatie op de volgorde van data. Data die ver in de toekomst liggen worden geaccepteerd. Voor het aanpassen van een datum zal de taak eerst gesloten en daarna opnieuw geopend moeten worden.

Een interessante bevinding treedt op als bij de geplande startdatum een jaar wordt opgevoerd dat voor 1900 ligt. Als dit het geval is dan kan de geplande startdatum na het sluiten en opnieuw openen van de taak niet meer aangepast worden. Gedurende het bestuderen van deze bevinding komt het team erachter dat de applicatie een log bestand aanmaakt (in de map Documenten) waarin eventuele fouten gelogd worden. De poging tot het aanpassen van de geplande startdatum leidt tot de volgende foutmelding in de log: "ValueError: year=1853 is before 1900; the datetime strftime() methods require year >= 1900". Na deze bevinding wordt er verder gekeken naar andere functionaliteit rondom tijd en datum. Zo wordt er een herinnering op een taak ingesteld op 5 minuten. De herinnering werkt.

TaskCoach bevat naast de planning van taken ook functionaliteit om de tijdsbesteding per taak bij te houden. Na enig onderzoek blijkt dat deze functionaliteit complex is en dat niet makkelijk te achterhalen is hoe TaskCoach precies omgaat met tijdsbesteding. Het bijhouden van tijdsbesteding kan gestart worden met een knop, maar kan ook gedaan worden door het opvoeren van de gespendeerde tijd in een apart tabblad. Het team doet de bevinding dat bij het opvoeren van een besteding van een uur deze besteding getoond wordt als slechts enkele seconden op het tabblad. Het is mogelijk om de bestedingen te aggregeren op maandniveau. Als we dit doen zien we dat de beschrijvingen die we voor elke besteding hebben opgevoerd bij elkaar worden gevoegd in één tekstveld.

De tijd die een applicatie gebruikt is afhankelijk van de tijdinstelling op het systeem. Het team manipuleert om die reden de systeemtijd van een MacBook Pro. De kalender wordt aangepast naar een Koptische kalender. Deze aanpassing zorgt ervoor dat de applicatie na opstarten vastloopt en dat in de log een melding verschijnt over een niet valide datum formaat.

Tot slot kijkt het team naar de samenhang tussen de gebudgetteerde tijd en de bestede tijd. TaskCoach is in staat om de resterende tijd te berekenen aan de hand van de genoemde variabelen. Er worden enkele testen uitgevoerd met variaties in uren, minuten en seconden. TaskCoach handelt al deze variaties correct af.

Terugkoppeling

Een teamlid brengt mondeling verslag uit over de sessie aan een teamlid van het andere team. Uit de terugkoppeling blijkt dat dit verslag veel detailinformatie bevat. Om de detailinformatie te kunnen plaatsen is een raamwerk nodig. De productschets had gebruikt kunnen worden om dit raamwerk in te vullen. Uit de terugkoppeling ontstaan één nieuwe charter, namelijk het testen van de synchronisatie van taken tussen systemen met verschillende systeemtijden.

Conclusie

Met het afronden van de test sessies wordt de bijeenkomst afgesloten. Terugkijkend hebben we In een dag met twee teams gericht een aantal testen uitgevoerd tegen een applicatie die vooraf onbekend was. We hebben laten zien dat er technieken en methoden bestaan die de tester helpen bij het opdoen van kennis over de applicatie, het ontwikkelen van een strategie en het uitvoeren van testen. Het verkennen van de applicatie levert inzichten op die als uitgangspunt dienen voor risico inschattingen en voor het uitvoeren van test sessies met een vastomlijnd doel. Door snel met concrete en onderbouwde testen te komen levert de tester in een kort tijdbestek waardevolle feedback over de applicatie. De test sessies worden geëvalueerd en bieden vervolgens aanknopingspunten voor verdere verdieping waar nodig.

Onder andere door de populariteit van Agile werkwijzen komt het regelmatig voor dat de tester gevraagd wordt iets te testen terwijl hij op dat moment onvolledig inzicht heeft in de functionaliteit van de applicatie die hij onderhanden heeft. Bovendien wordt van de tester verwacht dat hij binnen beperkte tijd resultaat levert. Het vraagt om een aanpak waarin de tester snel een strategie opstelt en uitvoert door te modeleren, kritisch te denken, te ontdekken en te onderzoeken. Dit is de aanpak die we hebben toegepast tijdens de hierboven beschreven bijeenkomst.

Verwijzingen

Bach, J. (1996). Heuristic Test Strategy Model.

Kaner, C., Bach, J., & Pettichord, B. (2001). *Lessons Learned in Software Testing.* John Wiley & Sons.

Task Coach. (2018). Opgehaald van Task Coach: http://taskcoach.org/