Ohjelmistotestaajan ammatillinen kehitys testiautomaatiokehittäjäksi

Tero Tallgren

2021 Laurea

Laurea-ammattikorkeakoulu

Ohjelmistotestaajan ammatillinen kehitys testiautomaatiokehittäjäksi

Tero Tallgren

Tietojenkäsittely

Opinnäytetyö

Marraskuu, 2021

Laurea-ammattikorkeakoulu **Tiivistelmä**

Tietojenkäsittely

Tradenomi (AMK)

Tero Tallgren

Ohjelmistotestaajan ammatillinen kehitys testiautomaatiokehittäjäksi

Vuosi 2021 Sivumäärä 40

Tämän opinnäytetyön tavoitteena oli selvittää toimivat ensi askeleet testiautomaatiokehitykseen sellaisen manuaalisen ohjelmistotestaajan näkökulmasta, jolla ei ole merkittävää ohjelmointikokemusta. Opinnäytetyössä toimeksiantajana toimi Vala Group Oy.

Opinnäytetyössä käsitellään testauksen ja testiautomaation periaatteita sekä näiden merkitystä ohjelmistokehitykselle. Kyseessä on laadullinen tutkimus, jossa käytettiin aineiston hankintamenetelminä teemahaastatteluja ja havainnointia. Näiden lisäksi opinnäytetyön aikana pidettiin Vala Group Oy:n sisäinen työpaja opinnäytetyön testiautomaatiokehittäjien kesken aiheesta sisältäen esityksiä, ryhmähaastattelun sekä kyselyn. Aineistona hyödynnettiin myös aikaisempaa yrityksen sisäistä kyselyä liittyen testiautomaatiotyökaluihin. Kerätty aineisto sisällytettiin opinnäytetyöhön mahdollisimman kattavasti, jotta lukijallakin on mahdollisuus tutustua tarkemmin lähdemateriaaliin.

Aineiston perusteella etenkin Suomessa Robot Framework on hyvin yleinen ja suosittu testiautomaatiotyökalu. Tulosten pohjalta luotiin johtopäätökset sopivista ensi askeleista testiautomaatiokehittäjäksi muutamassa eri tilanteessa. Tutkimuksen tuloksia voidaan hyödyntää ohjelmistotestaajien urasuunnittelussa testiautomaatiokehittäjiksi sekä yrityksen oman intranet testiautomaatio-ohjeistuksen päivitykseen.

Asiasanat: ohjelmistotestaaja, testiautomaatio, testaus, urapolku

Laurea University of Applied Sciences **Abstract**

Bachelor´s Degree Programme in Business Information Technology

Bachelor of Business Administration (BBA)

Tero Tallgren

Software Tester's Professional Development into Test Automation

Year 2021 Pages 40

The objective of this Bachelor's thesis was to find out the practical first steps into test automation development from the perspective of a manual software tester without extensive programming knowledge. Vala Group Ltd was the commissioner of this thesis.

The thesis covers the principles of software testing and test automation and their significance in software development. This thesis is a qualitative research where semi-structured interviews and observation were used as research methods. An internal workshop held during the thesis process included two expert presentations, a group interview and a survey. Also a previously made internal survey about test automation tools was utilized as research material. Gathered research material is presented extensively in the thesis for the benefit of the readers.

Based on the material, Robot Framework is a widely used and popular test automation tool in Finland. Conclusions on the first steps to test automation in a couple of different scenarios were presented in the end. The results of this thesis can be utilized in forming career paths to test automation and in updating the commissioner's intranet test automation instruction pages.

Keywords: Software tester, test automation, testing, career path

Sisällys

1 Johdanto

Testauksen ja etenkin testiautomaation rooli nykyajan ohjelmistokehityksessä kasvaa koko ajan yritysten pyrkiessä jatkuvasti nopeampaan ja virheettömämpään sovellusten julkaisuun. Tämä asettaa uusia odotuksia myös ohjelmistotestaajille. Testaajien odotetaan osallistuvan testiautomaation rakentamiseen ja edelleen suorittavan myös manuaalista testaustyötä yhtä laadukkaasti kuin ennenkin sovellusten julkaisuaikataulujen kärsimättä, mikä käytännössä edellyttää testiautomaation rakentamista hoitamaan etenkin regressiotestausta. Testiautomaatiolla ei voida kokonaan korvata manuaalista testausta, vaan se on noussut tärkeäksi työkaluksi ohjelmistojen laadunvarmistuksessa, mahdollistaen testaajien ajankäytön regressiotestauksen sijasta esimerkiksi testauksen suunnitteluun ja tutkivaan testaamiseen.

Tämän opinnäytetyön tarkoituksena on kartoittaa ohjelmistotestaajien ammatillista kehityspolkua testausautomaatiokehittäjiksi manuaalisen ohjelmistotestaajan näkökulmasta ja lähtökohdista tilanteessa, jossa aikaisempaa merkittävää kokemusta ohjelmoinnista sovelluskehittäjänä ei ole. Tutkimuksen ensisijaisena tavoitteena on muodostaa käsitys siitä, mitä testiautomaatio-osaajaksi kehittyminen vaatii, millä keinoin oppiminen lähtee hyvin käyntiin ja mihin asioihin testiautomaation oppimisprosessissa kannattaa keskittyä. Mitkä ovat toimivia ensiaskeleita, joista kannattaa lähteä liikkeelle ja löytyykö päämäärän saavuttamiseen kenties useampiakin vaihtoehtoja? Millaista tukea oppimisprosessi edellyttää ja onko sitä helposti saatavilla?

Tutkimustuloksia hyödyntäen voin myöhemmin itse perehtyä testausautomaatiokehittämiseen työni ohessa. Tutkimuksen tuloksia voidaan hyödyntää myös yrityksen nykyisen testiautomaation itseopiskeluaineiston päivitykseen ja ottaa huomioon muussa testiautomaatio-osaamisen laajentamiseen tähtäävässä toiminnassa.

2 Opinnäytetyön lähtökohdat

Lähtökohtana opinnäytetyölle oli omakohtainen tarpeeni opetella testiautomaatiota seuraavana ura-askeleena kohti kokonaisvaltaisempaa testauksen ammattilaisuutta. Olen usean vuoden ajan toiminut ohjelmistotestaajana tehden manuaalista testaustyötä, alkaen ensimmäisistä testausprojekteista yli kymmenen vuoden takaa. Testauksen perustason sertifikaatin olen suorittanut vuonna 2018. Oman urakehityksen kannalta seuraava looginen askel on testiautomaatiokehittämisen haltuunotto. Tässä opinnäytetyössä halusin selvittää, miten muut vastaavassa tilanteessa olleet ohjelmistotestaajat etenkin omalla työnantajallani Vala Group Oy:llä ovat testausautomaatioon perehtyneet ja sen omaksuneet.

Testauksen automatisoinnilla voidaan monin paikoin parantaa testauksen luotettavuutta, nopeutta sekä kattavuutta, jolloin ohjelmistotestaajat voivat toistuvien, rutiininomaisten testien sijaan keskittyä enemmän esimerkiksi tutkivaan testaukseen ja luovempiin testitapauksiin. Näin ollen ohjelmistotestauksen ammattilaisilta yhä enenevissä määrin odotetaan myös testiautomaatio-osaamista, kyvykkyys pelkästään laadukkaaseen manuaaliseen testaamiseen ei monin paikoin enää riitä.

Urakehitys manuaalisesta ohjelmistotestauksesta testiautomaation pariin on varsin yleistä ohjelmistotestaajilla. Kuitenkaan selvää ammatillista, dokumentoitua kehityspolkua testiautomaatiokehittäjäksi ei ollut saatavilla, vaikka testiautomaatio-osaamiselle on nykyään paljon kysyntää. Tutkimukselle nähtiin myös työnantajayrityksessä tarvetta, sillä yrityksessä on muitakin samassa tilanteessa olevia työntekijöitä ja oletettavasti on jatkossakin. Toisaalta yrityksessä on monia testiautomaation omaksuneita työntekijöitä, joiden kokemuksia ja osaamista voidaan hyödyntää tämän tutkimuksen kuluessa.

Arviolta noin 50 prosenttia Valan testauspalveluiden kysynnästä kohdistuu tällä hetkellä testiautomaatioon, 30 prosenttia testauksen johtamisrooleihin ja ainoastaan 20 prosenttia tavanomaiseen manuaaliseen testaustyöhön, tämän osuuden ollessa edelleen vähenemään päin. Maailmanlaajuisen testiautomaatiomarkkinan ennustetaan yli kaksinkertaistuvan vuoden 2019 tasosta eli 12,6 miljardista dollarista 28,8 miljardiin dollariin vuoteen 2024 mennessä (MarketsAndMarkets 2019).

2.1 Aihealueen rajaus

Tutkimuksen teoreettisena viitekehyksenä toimivat ohjelmistotestauksen ja testiautomaation periaatteet sekä näiden merkitys ohjelmistokehityksessä. Opinnäytetyössä ei käydä lävitse ohjelmistotestauksen osalta testauksen suunnittelua, testaustyön etenemistä tai eri testausmenetelmiä. Tutkimuksessa keskitytään aloittavan testiautomaatiokehittäjän kannalta olennaisiin ensiaskeliin, joten testiautomaatiokehittämisen teknisiä yksityiskohtia tai työkaluja ei käydä opinnäytetyössä kattavasti lävitse.

Oppimisprosessin ensiaskeleissa ja testiautomaation työkalujen osalta opinnäytetyössä keskitytään niihin, mitkä ovat Suomen markkinoilla merkityksellisiä testiautomaatiokehittäjille. Ulkomailla käytettävät testiautomaation työkalut voivat erota merkittävästikin siitä, mitä Suomessa käytetään.

2.2 Keskeisiä käsitteitä

Regressiotestaus

Uusien muutosten jälkeen toteutettavaa järjestelmän uudelleen testausta pyrkimyksenä varmistaa, että uudessa versiossa mikään aikaisemmin toiminut ei ole mennyt rikki uusien muutoksen takia.

Tuotantoympäristö

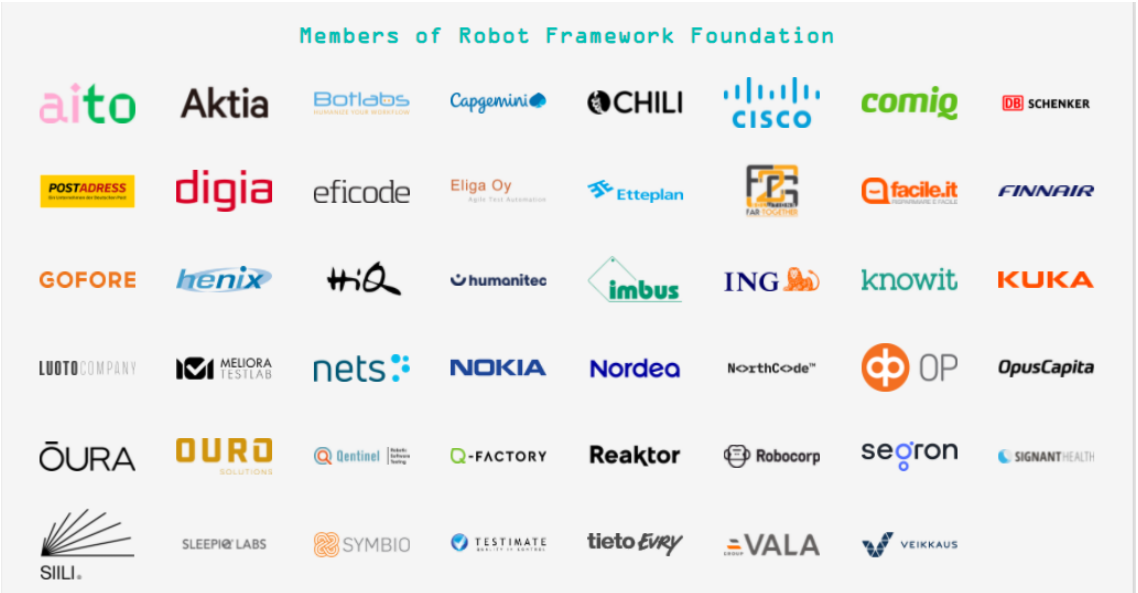
Sovelluksen loppukäyttäjien käytössä oleva julkaistu sovellusversio.

Testiympäristö

Tuotantoympäristöä jäljittelevä käyttöympäristö, jossa sovellusta ja sen ominaisuuksia testataan ennen tuotantoon siirtoa.

Robot Framework

Python ohjelmointikielellä toteutettu avoimen lähdekoodin ilmainen testiautomaatiotyökalu, jonka käyttö ei vaadi syvää ohjelmointitaitoa. Robot Framework on laajalti käytössä Suomessa, helposti integroitavissa muiden työkalujen kanssa, aktiivisesti tuettu ja sillä on laaja ekosysteemi ja testiautomaatiokehittäjien yhteisö ympärillään. Robot Frameworkin kehitystä rahoittavat useat yritykset (kuvio 1), jotka haluavat varmistaa sen jatkuvuuden. (Robot Framework 2021.)



Kuvio 1: Robot Frameworkin rahoittajat (Robot Framework 2021)

3 Tutkimus- ja kehittämismenetelmät

Tutkimus toteutetaan yleensä joko määrällisiä tai laadullisia tutkimusmenetelmiä hyödyntäen, ja joissain tapauksissa monimenetelmätutkimuksena sekä laadullisia että määrällisiä menetelmiä käyttäen. Erilaisten tutkimusmenetelmien, näkökulmien ja aineistojen hyödyntämistä samassa tutkimuksessa kutsutaan triangulaatioksi. Sen avulla voidaan esimerkiksi pyrkiä parantamaan tutkimuksen kattavuutta ja luotettavuutta eri menetelmien ja aineistotyyppien täydentäessä toisiaan. Tutkimus voidaan esimerkiksi aloittaa kyselyllä ja sen jälkeen käyttää haastatteluita syventämään tutkijan kokonaiskuvaa tutkimuskohteesta. (Eskola & Suoranta 2008, 68-70, 73; Vilkka 2020, 70-72.)

Määrällinen tutkimus pyrkii objektiivisesti, eli tutkijasta ja muista aikaisemmista näkökulmista tai materiaaleista riippumatta, lukujen perusteella kuvaamaan yleisellä tasolla tutkittavaa ilmiötä. Tällöin käytettävän tutkimusmateriaalin laajuudella ja kattavuudella on suuri merkitys peilattaessa tutkimuksen luotettavuutta. Mahdolliset yksittäiset poikkeavuudet suodattuvat tällöin pois, kun pyritään tekemään tutkittavasta ilmiöstä johtopäätöksiä suuremman massan perusteella. (Juuti & Puusa 2020, 75; Vilkka 2020, 66-67.)

Tämä opinnäytetyö on laadullinen tutkimus, jossa pääasiallisena tietolähteenä toimivat haastattelut sekä kysely rajatulle joukolle. Tutkimuksessa on käytetty lisäksi havainnointia tukemaan tutkijan ymmärrystä testiautomaatiokehittämisestä käytännön tasolla.

Laadullisen tutkimuksen keinoin voidaan pyrkiä syvällisesti ymmärtämään, hankkia uutta tietoa, kuvaamaan tai tulkitsemaan jotain ilmiötä. Laadullista tutkimusta voidaan pitää enemmän subjektiivisena tutkijan omien näkemysten, toimenpiteiden ja valintojen vaikuttaessa tutkimukseen koko sen elinkaaren ajan aina suunnitteluvaiheesta lopullisiin tuloksiin ja johtopäätöksiin. Laadullisen tutkimuksen aineisto tyypillisesti perustuu ihmisten subjektiivisten kokemuksiin ja niiden analysointiin. (Juuti & Puusa 2020, 59, 77; Vilkka 2020, 67-68.)

Määrällinen tutkimus etenee tyypillisesti lineaarisemmin vaiheesta toiseen. Sen sijaan laadullinen tutkimus ei välttämättä etene suoraviivaisesti alusta loppuun, vaan on tutkijan kannalta joustavampi tutkimussuunnitelman ja sen toteutuksen osalta. Laadullinen tutkimus voi edetä vaiheesta toiseen ja takaisin ymmärryksen ja aineiston kertyessä. Tutkimussuunnitelmaan, tutkimuskysymyksiin, tutkimuksen tavoitteisiin, sen rajaukseen sekä jo aiemmin tehtyihin ratkaisuihin tai oletuksiin voidaan palata prosessin eri vaiheissa ja tarvittaessa tehdä muutoksia niihin uuden tiedon ja ymmärryksen valossa. Tätä vähitellen oikeisiin johtopäätöksiin pyrkimistä vaiheesta toiseen edestakaisin liikkumalla kutsutaan hermeneuttiseksi kehäksi. Tutkijalla on ennen tutkimusta tietyt ennakko-oletukset tutkittavasta ilmiöstä ja jatkuvasti tutkimuksen edetessä hän korjaa omia näkemyksiään ja pyrkii vähitellen lähemmäksi aineiston pe-

rusteella kokonaisvaltaista ymmärrystä ilmiöstä ja aineistosta ja ilmiöstä tehtäviä oikeita johtopäätöksiä. (Eskola & Suoranta 2008, 15-16, 20; Juuti & Puusa 2020, 12-13, 73-74; Vilkka 2020, 66-67.)

Tyypillisesti laadullisessa tutkimuksessa aineistoa kerätään yksittäisiltä ihmisiltä tai ihmisistä erilaisin haastatteluin, havainnoinnein tai olemassa olevaa kirjallista dokumentaatiota kuten elämänkertoja tai lehtiartikkeleita hyödyntäen (Juuti & Puusa 2020, 13, 85). Laadullisella tutkimuksella ja laadullisen tutkimuksen keinoin kerättävän materiaalin avulla yleisesti pyritään ilmiön mittaamisen ja toistettavuuden sijasta syvällisemmin ymmärtämään tutkittavaa ilmiötä. Tutkimusmateriaalissa keskitytään määrän sijasta laatuun, sen huolelliseen ja rajattuun valintaan, jolla kuitenkin saadaan olennainen kattavuus ja ymmärrys tutkimuskohteesta. Tutkimukseen valitaan esimerkiksi haastateltaviksi tarpeeksi monta asiantuntevaa henkilöä, niin monta kuin on tarpeen tarvittavan aineistomäärän keräämiseksi. Aineiston määrä on siis tutkimuskohtaista ja pienemmän näytemäärän johdosta myös tutkimusaineistossa esiintyviin poikkeuksiin yleensä kiinnitetään enemmän huomiota, ja tarvittaessa hankitaan lisäaineistoa niiden analysointiin. Tutkijan on voitava myös perustella tekemänsä valinnat tutkimukseen valittujen näytteiden, kuten henkilöhaastattelujen suhteen, ja tuotava esille tutkimusprosessin aikainen edistyminen ilmiön ymmärtämisessä, jotta lukijat kykenevät seuraamaan ja arvioimaan tutkimuksen luotettavuutta. (Eskola & Suoranta 2008, 18, 62; Juuti & Puusa 2020, 14, 80, 84; Vilkka 2020, 67.)

Laadulliset tutkimusmenetelmät eivät kuitenkaan ole automaattisesti laadukkaan tutkimuksen tae. Tutkimuksen laatu ja lopulta sen hyödyllisyys riippuu aina tutkijasta. Hänen on osattava käyttää tutkimuskohteen kannalta sopivia tutkimusmenetelmiä, olivat ne sitten määrällisiä, laadullisia tai näiden yhdistelmä. (Vilkka 2020, 68.)

3.1 Kysely

Kysely on nopea ja tehokas keino kerätä monenlaista tietoa suurelta joukolta ihmisiä ja tavallisin keino kerätä aineistoa määrälliseen tutkimukseen. Kysely voidaan toteuttaa monella tapaa, esimerkiksi postitse, sähköpostitse, puhelimitse tai Internet-kyselynä. (Moilanen, Ojasalo & Ritalahti 2015, 121; Vilkka 2020, 94-95.)

Kyselyä laatiessa on olennaista, että tutkimuskohteesta on aiempaa tietoa saatavilla, jotta kysymykset voidaan suunnitella järkevästi. Kyselyssä tulisi olla ainoastaan sellaisia kysymyksiä, joilla on merkitystä ja jotka ovat olennaisia tutkimuksen kannalta. Kysymysten tulee olla ymmärrettäviä, selkeitä ja kysymyksessä tulisi selkeyden vuoksi kysyä vain yhdestä asiasta kerrallaan. Liian pitkä kysely ja huonot kysymykset vaikuttaa heikentäen kyselyn laatuun ja myös vastaajien vastaamishalukkuuteen. Kyselyssä vastaajille esitetään kaikille samat kysymykset samassa muodossa. Kysymyksiin voidaan liittää valmiit vastausvaihtoehdot, tai tilan-

teesta riippuen myös jättää avoimeksi vastaajan täytettäväksi hänen omilla sanoillaan. Avoimissa vastauksissa on vaarana, että niihin ei vastata mitään. Kyselyä voidaan käyttää myös arkaluontoisen materiaalin keräämiseen, vastaajan jäädessä aina tuntemattomaksi. (Moilanen, Ojasalo & Ritalahti 2015, 121, 130-132; Vilkka 2020, 94-95.)

3.2 Havainnointi

Havainnoinnissa tutkija tarkastelee suoraan todellisia tilanteita luonnollisessa toimintaympäristössään, mitä havainnoitava tekee, miksi ja miten. Havainnointia voidaan käyttää tutkimuksessa pääasiallisena menetelmänä tai toisen menetelmän, kuten haastattelun tai kyselyn, tukena. Havainnointi olisi hyvä etukäteen suunnitella tutkimussuunnitelmaan perustuen, mitä asioita havainnoidaan ja mihin kiinnitetään huomiota, ja havainnoinnin kuluessa järjestelmällisesti myös kirjata havainnot ylös. (Juuti & Puusa 2020, 131-132; Moilanen ym. 2015, 114; Vilkka 2020, 149.)

Tutkija voi suorittaa havainnointia itse osallisena ja vuorovaikutuksessa tutkimuskohteen kanssa, esimerkiksi työpaikalla suorittaen asiaan liittyvää tehtävää. Tällaista osallistuvaa havainnointia voidaan käyttää esimerkiksi hiljaisen tiedon keräämiseen ja edellyttää yhteisöön sisäänpääsyä tai sinne jo kuulumista. Tutkija voi myös pysyttäytyä ulkopuolisena tarkkailijana, pysyen ilmiön ulkopuolella vaikuttamatta tapahtumiin. Tämä voi sopia esimerkiksi tutkimuksen alkuvaiheessa, kun havainnoitava ilmiö tai henkilöt ovat vielä vieraita. Havainnoinnista ja tutkijan siinä ottamasta roolista tulisi etukäteen olla sovittu havainnoitavienkin kanssa, jos mahdollista. Joissain tilanteissa vieras havainnoija voi läsnäolollaan muuttaa tai häiritä asioiden luonnollista kulkua. Havainnointi on myös hyvin subjektiivista toimintaa, kaksi eri tutkijaa saattavat kiinnittää huomiota samassa tilanteessa aivan eri asioihin. (Eskola & Suoranta 2008, 102; Juuti & Puusa 2020, 132-133; Moilanen ym. 2015, 115-116; Vilkka 2020, 143-145.)

3.3 Haastattelu

Haastattelut ovat tutkijan aloitteesta käytyjä ohjattuja keskusteluja, joiden tavoitteena on saavuttaa lisätietoa tutkimuskohteesta (Puusa 2020, 103-104). Haastattelu on myös yleisin tiedonkeruumenetelmä laadullisissa tutkimuksissa (Puusa 2020, 103) ja joustavuutensa myötä soveltuvat hyvin tiedon hankintaan erilaisissa tilanteissa. Haastattelu voidaan toteuttaa kasvokkain, puhelimitse tai vaikka sähköpostitse tai pikaviestisovelluksen avulla ja haastateltavina voi olla yksi tai useampia henkilöitä, jolloin näkökulma on usein vähän erilainen. Yksilöä haastatellessa saadaan yksilön näkemys ja useampaa haastatellessa yhteisön näkemys. Haastattelu sopii hyvin tiedonsaantiin, kun olemassa olevaa tutkimustietoa on vähän, ja sitä voidaan usein käyttää muiden tutkimusmenetelmien kanssa niin, että ne tukevat toisiaan. (Moilanen ym. 2015, 106; Vilkka 2020, 123.)

Ryhmähaastattelussa haastatellaan useita henkilöitä kerralla, joten sitä voidaan pitää siten tehokkaana haastattelutapana. Ryhmähaastattelu sopii käytettäväksi, jos halutaan saada ryhmän yhteinen näkemys tutkimuskohteesta tai muuten monipuolisesti tietoa tietystä aiheesta usealta henkilöltä. Usein pyrkimyksenä on saada aikaan keskustelua haastateltavien välillä, ja mahdollisesti saada näin jopa enemmän ja monipuolisempaa tietoa aiheesta kuin pelkästään yksilöhaastatteluina olisi mahdollista saada. Haastateltavien tulisikin olla jokseenkin samankaltaisia, kuten vaikkapa kollegoita keskenään, jotta keskustelua voidaan käydä samoin käsittein kaikkien ymmärtäessä toisiaan. Haastateltavat voivat tukea, innostaa tai haastaa toisiaan keskusteluissaan aiheeseen liittyen, mikä voi tuottaa rikasta lisäinformaatiota. Tämä voi kuitenkin käytännössä olla haasteellista toteuttaa. Haastattelijan tai haastattelijoiden tulisi aktiivisesti pyrkiä rohkaisemaan osallistujia keskustelemaan keskenään, mikäli tätä toivotaan. (Eskola & Suoranta 2008, 94, 96-97; Puusa 2020, 115-116.)

Haastateltaviksi pyritään saamaan henkilöitä, joiden tiedetään omaavaan hyödyllistä tietoa tutkimuskohteesta. Haastattelutilanne on siinä mielessä myös joustava, että tutkija voi haastattelun aikana johdatella keskustelua tarvitsemaansa suuntaan ja toisaalta myös tarvittaessa pyytää haastateltavaa tarkentamaan vastauksiaan tai näkemyksiään. Olennaista haastatteluissa on pyrkiä saamaan mahdollisimman paljon tietoa haastateltavilta tutkimuskohteesta. (Puusa 2020, 106-107.)

Haastattelut ovat vuorovaikutuksellisia tilanteita, eli sekä haastattelija että haastateltava vaikuttavat haastattelun kulkuun sekä siitä saatuihin tietoihin (Eskola & Suoranta 2008, 85). Haastattelijan omalla olemuksella sekä tietopohjalla aiheesta on suuri vaikutus haastattelun kulkuun. Jotta haastattelu onnistuisi mahdollisimman hyvin, tulee tutkijan olla haastatteluun hyvin valmistautunut. Tutkijan tulee kyetä rakentamaan oman ennakkotietonsa pohjalta haastattelukysymykset tai aiheet niin, että myös haastateltavat ymmärtävät ne oikein. Haastattelutilanteessa molempien tulisi ymmärtää toisiaan ja tutkijan täytyy tarvittaessa välttää ammattislangia tai kapulakieltä muokaten omaa ulosantiaan niin, että haastattelutilanteessa puhutaan samalla kielellä, jolloin molemmat ymmärtävät käsiteltävät asiat samoin. Joskus voi olla hyödyllistä toimittaa etukäteen haastateltavalle haastattelukysymykset ja aiheet. Toisaalta liian tiukat raamit voivat myös olla haitaksi siinä mielessä, että haastateltava saattaa keskittyä ainoastaan etukäteen annettuihin kysymyksiin, jolloin mahdollisesti jotain olennaista tietoa tutkittavan ilmiön kannalta voi jäädä haastattelussa käsittelemättä, mikäli tutkija ei ole osannut tästä erikseen kysyä. (Puusa 2020, 106-108.)

Haastattelutilanteessa on tärkeää, että sen ilmapiiri on luottamuksellinen, jotta haastateltava kykenee vapautuneesti kertomaan käsiteltävästä aiheesta. Haastattelijan tulisi välttää kuulustelumaista otetta ja muistaa, että haastateltavaa saattaa jännittää haastattelutilanne. Haastatteluvastauksia tulkitessaan tutkijan on hyvä pitää mielessä myös ihmisten taipumus antaa mieluummin yleisesti hyväksyttäviä vastauksia kuin täysin omaa näkemystään vastaavia,

jos ne sattuvat poikkeamaan vallitsevasta yleisestä yhteiskunnallisesta linjasta. (Puusa 2020, 106-108.)

Haastattelu kannattaa tallentaa, jotta tutkija voi keskittyä haastattelutilanteeseen ja haastateltavaan sekä voi myöhemmin palata haastatteluun uudelleen, ja ehkä löytää vastauksista uusia näkökulmia, joita ei haastattelutilanteessa tullut ajatelleeksi. Haastattelut kirjoitetaan yleensä auki haastattelujen jälkeen. Haastattelujen avaamisen tai litteroinnin tarkkuus riippuu siitä, millaista aineistoa haastattelulla kerätään. Jos vain haastateltavien vastausten sisällöllä on väliä, niin silloin riittää yleiskielinen kirjaus haastattelusta. Jos sen sijaan on esimerkiksi tarkoitus samalla tulkita haastateltavaa tarkemmin, hänen äänenpainoaan, tunteita ja kehonkieltä niin silloin tarkempi ja sanatarkka litterointi tutkijan lisäkommentteineen on paikallaan. (Moilanen ym. 2015, 107.)

Haastattelut voidaan jakaa kolmeen tai neljään eri tyyppiin sen perusteella, kuinka avoimia haastattelutilanteet ovat ja kuinka tiukasti tutkija pitää haastattelun kulkua ohjaksissaan. Nämä tyypit ovat strukturoitu, puolistrukturoitu tai teemahaastattelu sekä avoin haastattelu.

3.3.1 Strukturoitu haastattelu

Strukturoitu haastattelu on eri haastattelutyypeistä kaikista tiukin muoto. Tutkija määrittää haastattelun kulun ja laatii haastattelukysymykset ja vastausvaihtoehdot ennakkoon teoriaan pohjautuen. Kysymykset esitetään haastateltaville aina samassa järjestyksessä ja haastateltavat valitsevat vastausvaihtoehdoista itselleen sopivimmat. Haastateltava saattaa tosin poiketa järjestyksestä, etenkin mikäli kyseessä on lomakemuotoinen haastattelu, jonka haastateltava täyttää itsekseen. Tällöin lomakkeella olisi hyvä mainita, että kysymyksiin toivotaan vastattavan järjestyksessä. Strukturoitu haastattelumuoto onkin hyvin vastaava kuin lomakekysely. Näiden välisenä isona erona voidaan pitää osallistujajoukkoa, sen rajausta ja vastausprosentteja. Siinä missä kysely toimitetaan tyypillisesti suurelle määrälle vastaajia, joista vain pieni osa yleensä vastaa, haastattelu voidaan kohdentaa rajatuille henkilöille, joilta kaivattua tietoa haetaan, ja jos haastatteluista on sovittu etukäteen, niin vastauksiakin voidaan odottaa osallistujilta hyvällä todennäköisyydellä. (Puusa 2020, 111.)

Strukturoitu haastattelu on käyttökelpoinen etenkin silloin, kun halutaan täsmällistä tietoa rajatusta aiheesta ja haastattelukysymykset voidaan tiivistää muutamaan, kolmesta kuuteen, kysymykseen. Strukturoitu muoto myös kaventaa tutkijan roolia haastatteluissa, mikä voi joissain tilanteissa olla toivottua. Mikäli tutkimuskohde on laajempi, niin silloin avoimemmat haastattelutyypit toimivat yleensä paremmin. (Puusa 2020, 111; Vilkka 2020, 123.)

3.3.2 Puolistrukturoitu haastattelu ja teemahaastattelu

Puolistrukturoitu haastattelu on strukturoitua haastattelua avoimempi muoto, jossa vastausvaihtoehtoja, kysymysten tarkkoja sanamuotoja ja niiden järjestystä ei ole lukittu valmiiksi strukturoidun haastattelun tapaan. Haastateltavat kertovat vastauksensa omilla sanoillaan ja mahdollisesti tuovat esiin tietoja, joita tiukemmalla kysymys- ja vastausasettelulla ei olisi saavutettu. (Moilanen ym. 2015, 108; Puusa 2020, 111-112.)

Puolistrukturoitua ja avointa haastattelua voidaan käyttää esimerkiksi silloin, kun halutaan syvempää tietoa tutkimuskohteesta, tehdä tulkintaa määrällisen tutkimuksen tuloksista, ymmärtää jotain ilmiötä tutkittavien näkökulmasta tai ennen määrällistä tutkimusta kerätä tarvittavaa taustatietoa tutkimuskohteesta tai ilmiöstä (Moilanen ym. 2015, 109).

Teemahaastattelu on puolistrukturoitu haastattelu, jossa keskitytään tiettyyn teemaan tai teemoihin. Haastattelijalla ei välttämättä ole tarkasti määriteltyjä kysymyksiä, mutta haastattelija kuitenkin pitää huolen, että aiotut teemat käydään haastattelussa lävitse (Eskola & Suoranta 2008, 86). Puolistrukturoitu haastattelu voi olla jotain muutakin kuin teemahaastattelu, mikäli haastattelussa ei keskitytä mihinkään tiettyihin teemoihin tai aihepiiriin. Usein käytännössä tutkimuksessa ja haastatteluissa keskitytään kuitenkin tiettyyn aiheeseen ja tällöin käytetään tai puhutaan yleisemmin teemahaastattelusta. (Puusa 2020, 112; Vilkka 2020, 123.)

Haastattelun tarkoituksena on syventää tutkijan ymmärrystä tutkimuskohteesta haastateltavien omakohtaisten kokemusten ja näkemysten avulla. Lähtökohtana teemahaastattelussa on, että tutkija on tutustunut ennalta käsiteltäviin keskusteluteemoihin, siihen liittyvään kirjallisuuteen ja muuhun ennakkotietoon sekä itse haastateltavaan ja hänen toimenkuvaansa ja ympäristöönsä, ja kykenee näin ollen keskustelemaan luontevasti haastateltavan kanssa. (Puusa 2020, 112-113.)

Kysymysten tai teemojen käsittelyjärjestys ei ole samalla tavoin lukittua teemahaastattelussa kuin se on strukturoidussa haastattelussa, haastateltavien tulisi antaa vapaasti puhua aiheesta haluamassaan järjestyksessä. Tutkijan tulisi kuitenkin ohjata keskustelua etukäteen suunniteltujen raamien mukaisesti ja esittää teemoihin liittyviä tarkentavia kysymyksiä. Vaikka teemahaastattelun pitäisi edetä vapaasti haastateltavan ehdoilla, niin tutkijan tulee kuitenkin huolehtia, että keskustelu pysyy asetettujen teemojen sisällä. Tutkijan rooli voi myös muodostua erilaiseksi eri haastateltavien kanssa. Teemoja ja kysymyksiä voidaan käsitellä eri järjestyksessä, laajuudessa, eri tavoin ja erilaisin sanankäänteen. Välillä tutkija on lähinnä kuuntelijan roolissa ja välillä taas tarvitaan aktiivisempaa otetta suorine kysymyksineen. (Puusa 2020, 112-113; Vilkka 2020, 124, 126.)

3.3.3 Avoin haastattelu

Avoin haastattelu on tiettyyn määriteltyyn aiheeseen keskittyvä keskustelunomainen haastattelu, jossa haastattelija ja haastateltavat keskustelevat aiheesta vapaasti ilman etukäteen mietittyjä valmiita kysymyksiä tai teemoja strukturoidumpien haastattelutyyppien tapaan. Haastateltava ohjaa tilannetta eteenpäin omilla kertomuksillaan, johon haastattelija reagoi omilla jatkokysymyksillään. Jos haastateltavia on useita, niin eri haastateltavien kanssa käydyt keskustelut voivat olla hyvin erilaisia ja eri teemoihin liittyviä. (Eskola & Suoranta 2008, 86; Puusa 2020, 114; Vilkka 2020, 127.)

Yksi avoimen haastattelun muodoista on syvähaastattelu, jossa pyritään mahdollisimman hyvin avaamaan haastateltavan kanssa käsiteltävää aihetta tai tutkimuskohdetta, useiden keskustelukertojen ajan. Haastattelija esittää tarkentavia ja uusia kysymyksiä haastateltavan kertoman perusteella vieden näin keskustelua eteenpäin. Syvähaastattelu edellyttää haastattelijalta vahvaa tietämystä aihealueesta ja tyypillisesti haastateltavia on vain yksi tai muutama. (Puusa 2020, 114; Vilkka 2020, 127.)

4 Ohjelmistotestaus ja sen merkitys ohjelmistokehityksessä

Yksinkertaisesti ohjelmistotestauksen tarkoitus on varmistaa, että kehitettävä sovellus vastaa sille asetettuja odotuksia ja toimii kuten sen on suunniteltu toimivan, täyttäen asiakkaan sille asettamat tarpeet. Testauksen tavoitteena on löytää mahdolliset viat ja puutteet ohjelmiston kehitysvaiheessa ennen tuotteen julkaisua, mikä johtaa ohjelmiston parempaan laatuun. Testaus on yleensä ohjelmistoprojektin kallein yksittäin osakokonaisuus, esimerkiksi Suomessa testauksen osuus on noin kolmannes ohjelmistotalojen kehitysbudjetista. Testaus itsessään ei luo lisäarvoa, mutta on silti välttämätön osa laadukkaan ohjelmiston kehityksessä. Hyvin toteutettuna testaus säästää projektissa aikaa ja kustannuksia. Huonosti toteutettu testaus tai jopa sen puute kokonaan ja siten toimimaton tai huonosti toimiva sovellus voi tulla hyvin kalliiksi niin rahallisesti kuin yhtiön maineen osalta, ja myös aiheuttaa asiakaskatoa. Esimerkiksi Facebookin maailmanlaajuinen usean tunnin pituinen katko 4.10.2021 maksoi yritykselle menetettyinä mainostuloina Fortune-lehden arvion mukaan lähes 100 miljoonaa dollaria ja yrityksen osakekurssi laski samana päivänä lähes 5 %, mikä tarkoitti Facebookin omistajille kymmenien miljardien dollarien laskennallista tappiota (Matney 2021; Morris 2021). Silti testaus on usein se osa-alue, joka jää ohjelmistoprojekteissa viimeiseksi ja jonka laadusta ja kestosta ensimmäiseksi tingitään aikataulu- ja budjettipaineiden alla, vaikka itseasiassa ohjelmiston kannattavuuden kannalta onnistunut testaus on kaikista tärkein osuus kehitystyöstä. (Kasurinen 2013, 10-13, 16; Ratilainen 2019.). Laadukkaan ohjelmistotestauksen merkitystä ei usein

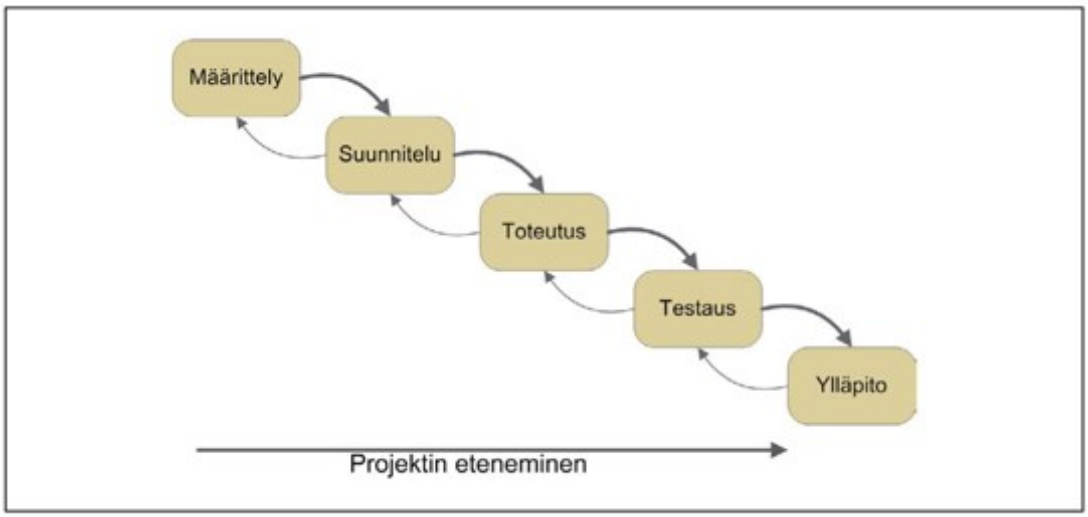
ymmärretä, jolloin se saatetaan alimitoittaa sekä jättää sille projektin loppuun vain se aika, mitä nyt sattuu jäämään jäljelle ennen ennalta päätettyä julkaisuaikaa.

Sovelluksen testaus alkaa ihanteellisesti jo vaatimusmäärittely- ja suunnitteluvaiheessa jatkuen läpi koko kehitysprosessin, kunnes suurimmat ongelmat on ratkaistu ja sovellus on valmis käyttöönotettavaksi. Tämän jälkeen siirrytään ylläpitovaiheeseen, jonka aikana myös yleensä tehdään korjauksia ja jonkinlaista jatkokehitystyötä, jotka vaativat testausta samalla tavoin kuin kehitysvaihe. (Kasurinen 2013, 13.) Jos testaus aloitetaan jo vaatimusten ja määrittelyjen katselmoinneilla, vähennetään riskejä sille, että lähdetään toteuttamaan kelvottomia ominaisuuksia ja koodia sekä sellaista ohjelmistoa, joka ei vastaakaan asiakastarpeita (ISTQB 2018, 13).

Testauksen osalta on hyvä kuitenkin huomioida, että sovelluksen täydellisen kattava testaus ei ole lähes koskaan mahdollista, ellei kyseessä ole hyvin pieni korjaus tai uudistus sisältäen vain vähän mahdollisia muuttujia. Testaajan ammattitaitoon kuuluu valita sopivat testaustavat ja testitapausmäärät kussakin tilanteessa ottaen huomioon testattavan osion merkitys kokonaiskuvassa ja testaukseen käytettävissä oleva aika. Testaajan tulee priorisoida omaa tekemistään ja testausta niin, että testaustyöllä saadaan paras mahdollinen kattavuus käytettävissä olevan ajan kuluessa.

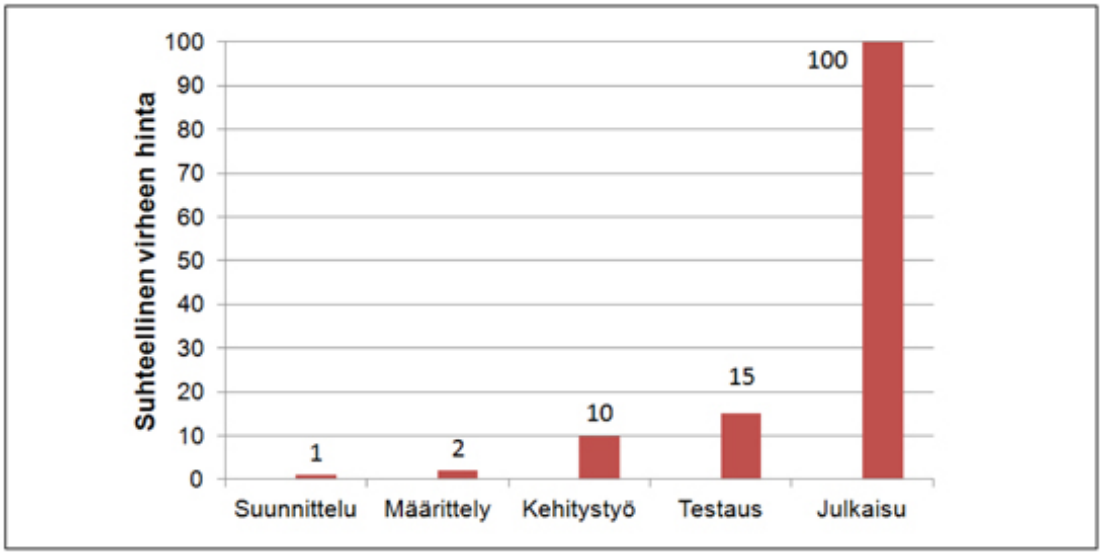
Onnistunut testaus edellyttää myös sitä, että testaaja ymmärtää, mikä projektissa on olennaista, mihin asioihin tulee keskittyä ja millaisia ovat asiakkaan tai sidosryhmien odotukset tuotoksen ja sen laadun suhteen. Testauksen prioriteetit ja käytettävissä olevat resurssit vaihtelevat paljon eri ohjelmistoprojekteissa. Joskus halutaan keskittyä vain kriittisten ongelmien löytämiseen ja näiden ratkaisuun ja toisaalla taas voidaan odottaa mahdollisimman virheetöntä lopputulosta. (Ratilainen 2019.)

Ohjelmistokehityksen vesiputousmallissa (kuvio 2) ohjelmistoprojekti etenee vaiheittain määrittelystä, suunnittelusta, toteutuksesta testaukseen, käyttöönottoon ja lopulta ylläpitovaiheeseen. Testaus on vain yksi vaihe kehitystyön ja käyttöönoton välissä tapahtuen siis vasta kaiken kehitystyön päätteeksi. Mallia pidetään vanhanaikaisena, koska esimerkiksi vaatimusmäärittelyn tekeminen kokonaan projektin alussa on hyvin haastavaa. (ISTQB 2018, 25; Kasurinen 2013, 12-14.)



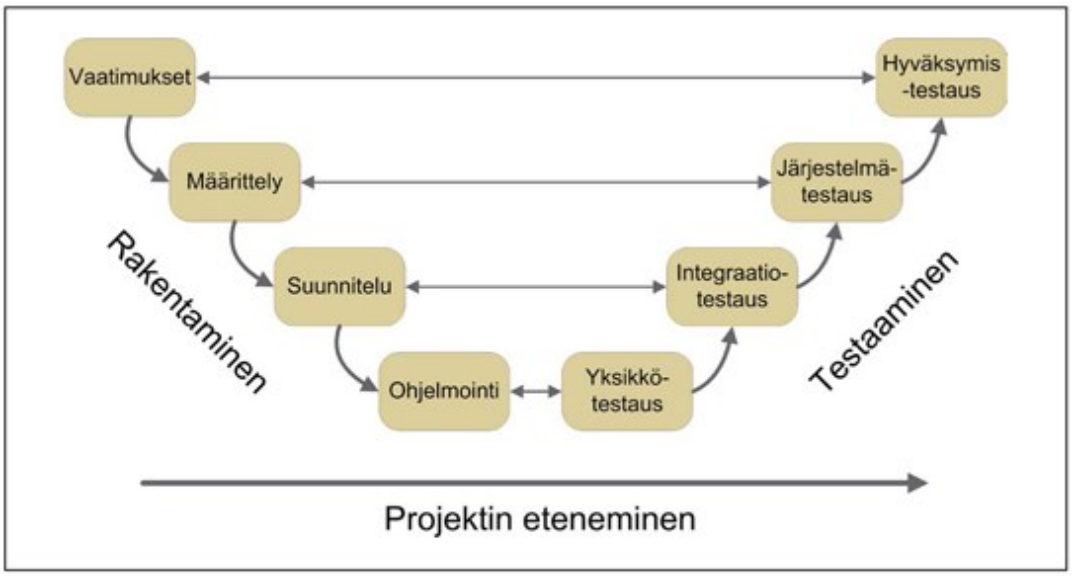
Kuvio 2: Vesiputousmalli (Kasurinen 2013, 13)

Harvoin kaikki sovellukseen vaikuttavat asiat ja odotukset on tiedossa heti projektin alussa, vaan siihen kohdistuvat odotukset ja suunnitelmat yleensä kehittyvät projektin kuluessa. Myös testauksen jättäminen pelkästään projektin loppuun on monelta osin ongelmallista ja haasteellista. Yleisesti ottaen suurin osa löydetyistä virheistä kohdistuu määrityksiin, ei itse tekniseen toteutukseen. Mikäli virheen havaitseminen olisi tapahtunut jo määrittelyn katselmoinnissa, niin projektissa olisi säästetty paljon rahaa ja aikaa. Suunnitteluvaiheessa havaitun virheen korjaaminen maksaa 1-2 prosenttia siitä (kuvio 3), mitä korjaus tulee maksamaan sovelluksen käyttöönoton jälkeen. (Kasurinen 2013, 12.)



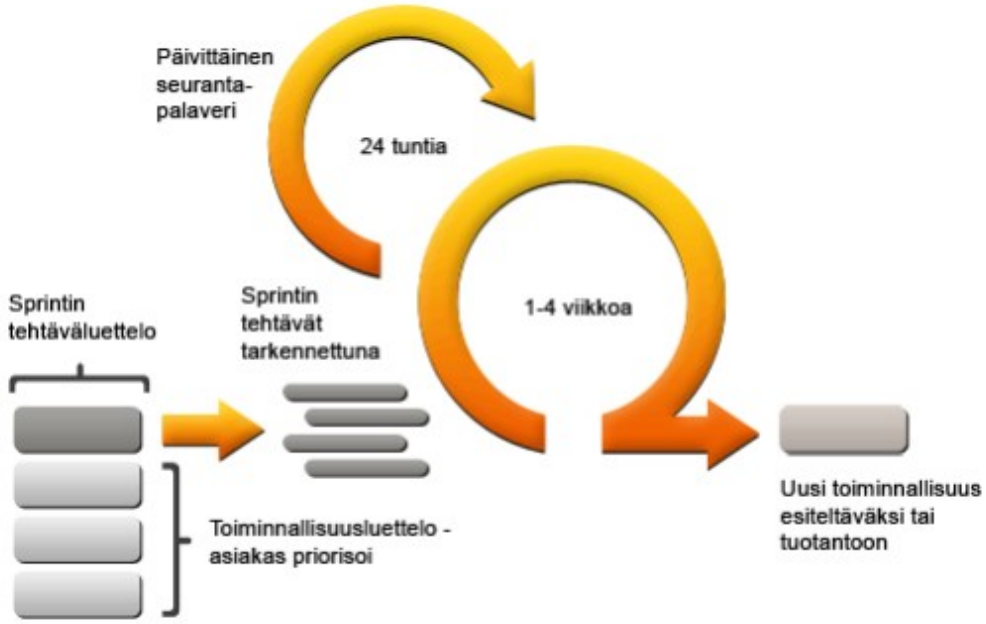
Kuvio 3: Testauksen kustannuskäyrä (Kasurinen 2013, 18)

Vesiputousmallista jatkokehitetyssä V-mallissa (kuvio 4) jokaisella kehitysvaiheella on sitä vastaava testausvaiheensa (Kasurinen 2013, 14). Tässäkin mallissa kuitenkin voi olla vesiputousmallin tapaan ongelmana se, että esimerkiksi asiakasvaatimusten ja määrittelyiden oikeellisuuden todentaminen voi jäädä vasta projektin loppuvaiheeseen, kun sovellus on jo lähes valmiiksi koodattu. Testaus tulisikin aloittaa jo ennen ohjelmointityötä vaatimusten ja määrittelyjen katselmoinneilla, mahdollisesti myös prototyyppien, eli nopeasti rakennettujen kokeilumallien, testauksella.



Kuvio 4: V-malli (Kasurinen 2013, 14)

Vesiputousmallin jäykkyyteen ja testauksen myöhäisen alkamisen ongelmiin ratkaisuna pidetään usein ketteriä ohjelmistokehitysmenetelmiä, kuten Scrum-mallia (kuvio 5). Siinä ideana on pilkkoa sovellus useaan pienempään osaan ja toteuttaa niitä yksitellen muutaman viikon pituisissa sprinteissä, kunnes sovellus on valmis. Näin mahdollisiin ongelmiin päästään vaatimuksia ja määrityksiäkin osalta aiemmin kiinni. Ketterien menetelmien avulla voidaan myös taklata projektiin tai sen asiakasodotuksiin liittyviä alkupään epävarmuuksia niin, että koko sovelluksen vaatimusmäärittelyn ja suunnittelun ei tarvitse olla jo ennen koodaustyötä kokonaan valmiina, vaan kehitystyö voidaan aloittaa yhdestä komponentista ja edetä seuraavien komponenttien vaatimusten, määrittelyjen ja suunnitelmien osalta samalla kuin ensimmäistä tai ensimmäisiä komponentteja rakennetaan ja testataan.



Kuvio 5: Scrum-prosessi (Juselius 2012, 14)

ISTQB, International Software Testing Qualifications Board, joka vastaa globaalisti ohjelmistotestauksen sertifikaattikokeiden järjestämisestä ja sertifikaattien jakamisesta (ISTQB 2021), luettelee perustason testaussertifikaattimateriaalissaan seuraavat seitsemän ohjelmistotestauksen yleistä perusperiaatetta (ISTQB 2018, 15):

1. Testauksella voidaan osoittaa, että sovelluksessa on vikoja mutta ei sitä, että niitä ei olisi lainkaan.
2. Täydellinen testaus ei ole mahdollista, testaajan tulee käyttää oikeita testaustekniikoita ja arvioida riskitekijöitä testauksen priorisoinnissa.
3. Testaus tulee aloittaa mahdollisimman aikaisin.
4. Viat keskittyvät muutamiin osiin tai komponentteihin, ja testaus tulisi keskittää näihin.
5. Hyönteismyrkkyparadoksi – testitapauksia pitää päivittää projektin edetessä, koska jossain vaiheessa kaikki alkuvaiheen viat on korjattu eivätkä alkuperäiset testit enää löydä uusia virheitä.
6. Testaus tapahtuu eri tavoin eri olosuhteissa ja tilanteissa. Esimerkiksi verkkosivun testauksessa painotetaan eri asioita kuin auton ajotietokoneen tai lentokoneen autopilotin testauksessa.
7. Virheettömyys ei merkitse mitään, jos ohjelma ei vastaa käyttäjän odotuksia ja tarpeita

4.1 Testaustasot

Edellisessä luvussa esitelty V-malli sisälsi neljä eri kehitysvaihetta ja niitä vastaavat testausvaiheensa tai -tasonsa; yksikkötestaus, integraatiotestaus, järjestelmätestaus sekä hyväksymistestaus. Tässä luvussa kydään tarkemmin lävitse millaisia ne ovat.

4.1.1 Yksikkötestaus

Yksikkötestaus on V-mallin ohjelmointia vastaava testaustaso. Yksikkötestauksen tarkoituksena on varmistaa, että uusi kirjoitettu koodi tai ohjelmakomponentti toimii kuten on oletettu. Koska testi koskee vain yhtä ohjelman osaa eli juuri kirjoitettua funktiota, moduulia tai muuta koodinpätkää, puhutaan yksikkötestistä tai komponenttitestistä. Tyypillisesti testin kirjoittaa ja suorittaa koodia kirjoittava sovelluskehittäjä koodin valmistumisen jälkeen (ISTQB 2018, 28).

Testi kirjoitetaan yleensä niin, että siinä tarkistetaan komponentin reagointia oikeisiin ja vääriin syötteisiin. Virhetilanteissa sovelluskehittäjä voi reagoida ja korjata ratkaisuaan, ennen kuin se liitetään osaksi kokonaisuutta. Testien rakentamista helpottamaan voidaan joutua rakentamaan testikomponentteja simuloimaan sovelluksen eri komponenttien välistä vuorovaikutusta. (Kasurinen 2013, 51.) Jos testi on automatisoitu säännöllisesti toistuvaksi testiksi, niin sen avulla voidaan saada kiinni ja korjattua jonkin muutoksen johdosta hajonnut komponentti tai funktio, ennen kuin virhetilanne pääsee toteutumaan tuotantoympäristössä.

4.1.2 Integraatiotestaus

Integraatiotestaus on V-mallin suunnittelua vastaava testaustaso. Integraatiotesteissä uusi komponentti kytketään toimimaan sovelluksen muiden osien kanssa. Testeissä keskitytään sovelluksen eri komponenttien väliseen vuorovaikutukseen, mahdollisiin rajapintoihin ulkoisten palveluiden tai sovellusten kanssa pyrkien varmistamaan että, kokonaisuus toimii yhdessä ja edelleen niin kuin sen pitää. Integraatiotestejä voidaan luoda esimerkiksi testaamaan komponenttien välistä kommunikaatiota sekä yhteisen tietokannan käyttöä. (ISTQB 2018, 28-29; Kasurinen 54.) Ulkoisten palveluiden kanssa tiedonvälitys voi perustua APIen käyttöön, eli näitä testataan myös, jos niitä käytetään.

Integraatiotestaus ja testien luonti voi olla kehittäjien tai testaajien vastuulla, riippuen tilanteesta. Integraatiotestit ovat tyypillisesti yksikkötestien ohella hedelmällisiä automaatiotestauksen kohteita. Automatisoiduilla testeillä voidaan varmistaa, että uudet muutokset eivät huomaamatta riko komponenttien, sovellusten ja eri palveluiden välisiä yhteyksiä ja rajapintoja (ISTQB 2018, 28).

4.1.3 Järjestelmätestaus ja hyväksymistestaus

Järjestelmätestaus on V-mallin kolmas ja määrittelyjä vastaava testaustaso, jossa järjestelmää testataan yksikkö- ja integraatiotestien jälkeen sitä varten rakennetussa tuotantoympäristöä vastaavassa testiympäristössä kokonaisuutena testaajien toimesta. Järjestelmätestaus sisältää kaiken kokonaiselle järjestelmälle tehtävän testauksen sille parhaiten soveltuvine testaustekniikoineen. (ISTQB 2018, 32; Kasurinen 2013, 56-57)

Testausta suoritetaan tai olisi ainakin syytä suorittaa sekä toiminnallisella että ei-toiminnallisella tasolla. Sovellusta testataan eri toiminnallisuuksien ja määriteltyjen käyttötapausten osalta niiden alusta loppuun asti tavoitteena varmistaa sovelluksen laatu ja tarkoituksenmukaisuus. Järjestelmätestivaiheessa pitäisi löytää aikaisemmista vaiheista läpipäässeet epäloogisuudet ja viat estäen niiden valumisen hyväksymistestiin ja tuotantoon sekä varmistaa että ohjelmoitu sovellus toimii kokonaisuutena vastaten sille luotuja niin teknisiä määrityksiä kuin asiakkaiden ja käyttäjien vaatimuksia. (ISTQB 2018, 31-32.)

Hyväksymistestiin siirrytään järjestelmätestauksen jälkeen. Se on kehitys- ja testausprosessin viimeinen vaihe ennen sovelluksen julkaisua ja käyttöönottoa, jossa tarkoituksena on varmistaa, että sovellus täyttää asiakkaan sille asettamat vaatimukset ja se voidaan siten hyväksyä käytettäväksi. Hyväksymistestiin osallistuvat testaajien lisäksi tai heidän sijaansa asiakkaat tai sovelluksen tulevat käyttäjät. Sovellusta testataan kokonaisuutena järjestelmätestin tapaan lähes tai täysin valmiilla sovellusversiolla käyttäjän näkökulmasta tuotannossa tai lähes tuotantoa vastaavassa testiympäristössä. Kaupallisten sovellusten, kuten vaikka pelien osalta, nähdään usein potentiaalisille asiakkaille suunnattuja avoimia tai rajattuja alfa- ja betatestauksia, joiden tavoitteena on kerätä palautetta testatusta sovelluksesta. (ISTQB 2018, 32-33; Kasurinen 2013, 57.)

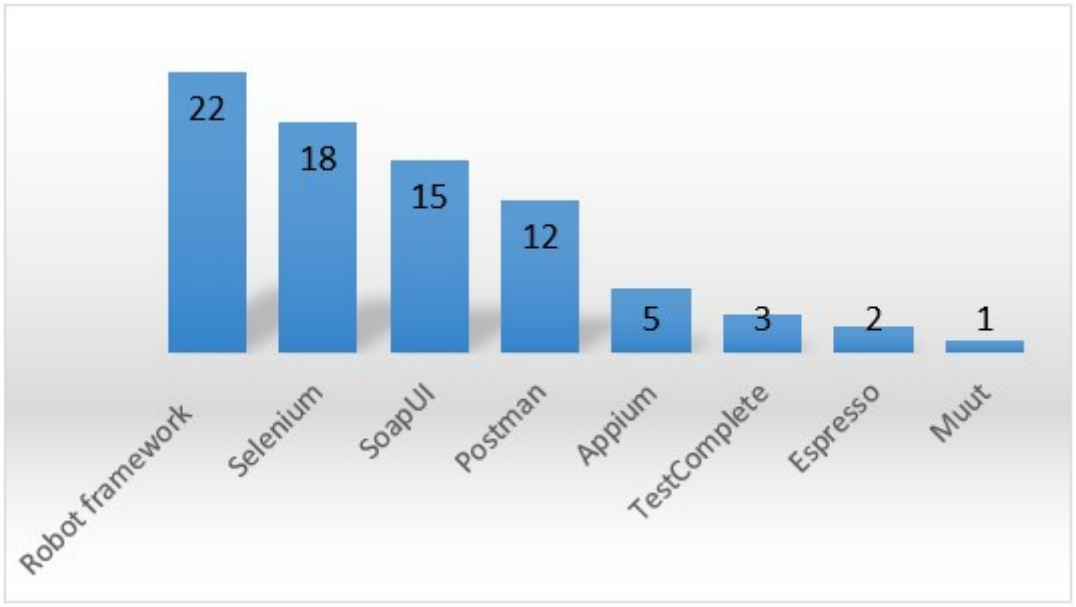
4.2 Testiautomaatio

Testiautomaatio tarkoittaa automaattista, koneellisesti tapahtuvaa testausta. Sen tarkoitus ei ole korvata käsin tehtävää testausta, vaan täydentää sitä, ja oikein toteutettuna testiautomaatio voi tuoda merkittäviä hyötyjä ohjelmistokehitykseen. Sovelluksen kehittyessä uusilla ominaisuuksilla myös sovelluksen regressiotestaustarve kasvaa samaan tahtiin, vieden enemmän ja enemmän testaajien aikaa. Toimiva testiautomaatio vapauttaa testaajien aikaa jatkuvasta regressiotestauksesta muihin tehtäviin, kuten testauksen parempaan suunnitteluun ja uusien toiminnallisuuksien testaamiseen. Automaattitestejä voidaan esimerkiksi ajaa joka yö, jolloin aamulla sovelluskehittäjät voivat reagoida mahdollisiin uusiin virheisiin. Automaattisia testejä voidaan rakentaa yksikkötestitasolla funktioille ja komponenteille, komponenttien ja sovellusten välisiin rajapintoihin sekä käyttöliittymän toimintoja varten. Joskus on järkevää tehdä testiautomaatiota näille kaikille tasoille, joskus vain osalle niistä. Testiautomaation

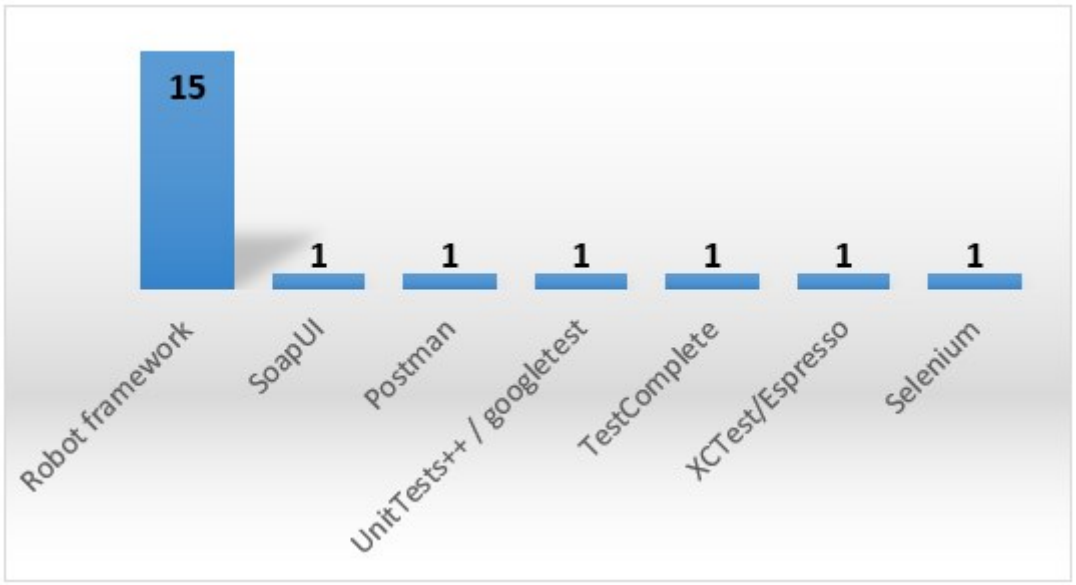
avulla ei yleensä pystytä löytämään uusia ongelmia, vaan sitä käytetään nimenomaan sovelluksen regressiotestaukseen varmistamaan sitä, että uusi sovellukseen lisätty koodi ei riko olemassa olevia aikaisempia toiminnallisuuksia. Testiautomaatio ei myöskään sovellu esimerkiksi käyttöliittymän ulkoasun ja käytettävyyden tai hyvin monimutkaisten ominaisuuksien testaamiseen. (Guberna 2020; Kasurinen 2013, 76-78.)

Testiautomaation rakentaminen ja ylläpito vaatii paljon resursseja, joten sitä suunnitellessa pitää huomioida, millaisia testejä on ylipäätänsä järkevää automatisoida. Testitapaus on yleensä järkevää automatisoida silloin kun sitä joudutaan toistamaan useasti, sen testaus on aikaa vievää, se sisältää paljon datavalidointia, testaus on ihmiselle tylsää tai liian vaikeaa. Ohjelmiston suorituskyky- ja stressitestaus on myös järkevää automatisoida, koska sitä ei pystytä tehokkaasti käsin tekemään. Mikäli testi toistuu vain harvoin tai sitä jouduttaisiin muuttamaan jatkuvasti, automatisointi vie enemmän resursseja kuin käsin testaus. Yhden testin rakentamiseen menee paljon aikaa ja on hitaampaa kuin sen suorittaminen käsin, mutta toisaalta kun automatisoitu testitapaus on valmis, sen uudelleen suoritus ei enää vie resursseja juuri lainkaan. Testiautomaatio ja sen edellyttämät projektiin sopivat työkalut vaativat erityisosaamista, jota ei välttämättä kaikilta testaajilta löydy. Testiautomaatio on luonteeltaan enemmän ohjelmistokehitystä kuin varsinaista testausta. Tehdyt testitapaukset ovat osa ohjelmakoodia ja haastavammat testitapaukset voivat vaatia edistynyttä ohjelmointiosaamista. Onnistunut testiautomaatiokehitys edellyttää yleensä projektin ohjelmistokehittäjienkin panosta. (Guberna 2020; Kasurinen 2013, 76-78.)

Sopivien työkalujen valinta on oleellista testiautomaation onnistumiselle. Työkalujen valinnassa tulee huomioida käytettävissä oleva budjetti, saatavilla olevat vaihtoehdot sekä kehitystiimin osaaminen. Avoimen lähdekoodin ratkaisut voivat joustavuutensa sekä ilmaisuutensa myötä soveltua projektiin paremmin kuin lisenssimaksulliset työkalut, toisaalta niiden käyttö voi vaatia myös enemmän osaamista. Mikäli mahdollista, työkaluja kannattaa testata käytännössä ennen lopullista valintaa. (Guberna 2020.) Valalla 2020 joulukuussa testiautomaatiokehittäjille suoritetussa kyselyssä tutuimpina testiautomaatiotyökaluina (kuvio 6) mainittiin käyttöliittymätestaukseen soveltuvat Robot Framework ja Selenium sekä API-testauksessa käytetyt SoapUI sekä Postman. Muutamia mainintoja saivat Appium, Testcomplete ja Espresso, ja näiden lisäksi mainittiin vielä lukuisia muita työkaluja kertaalleen. Mieluisimpana työkaluna (kuvio 7) korostui Robot Framework, muut työkalut saivat ainoastaan yksittäisiä mainintoja.



Kuvio 6: Kokemus testiautomaatiotyökaluista (Vala 2020)



Kuvio 7: Mieluisin testiautomaatiotyökalu (Vala 2020)

Testiautomaation etuna on testisuorituksen nopeus, tarkkuus ja luotettavuus. Kone ei koskaan unohda mitään testin vaihetta ja se suorittaa ohjelmoidun testin nopeammin kuin ihminen siihen kykenee. Testiautomaation avulla pystytään parantamaan testauksen kattavuutta sekä säästämään pitkässä juoksussa rahaa verrattuna manuaaliseen regressiotestaukseen. Testiautomaatio vapauttaa testaajien aikaa toistuvasta regressiotestauksesta mielekkäämpiin tehtäviin. (Guberna 2020.)

Testiautomaation haasteina voidaan pitää sen aikaa vievää käyttöönottoa, erillisiä työkaluja, resursseja ja erityisosaamista testaajilta. Myös vanhaa ohjelmistokoodiin voi olla hankalaa lähteä liittämään testiautomaatiota. Kun testejä on tehty, niin ne ovat etenkin käyttöliittymätestien osalta haavoittuvaisia. Testitapauksia tulee muistaa ylläpitää testien ollessa osa ohjelmakoodia, jolloin ohjelmakoodiin tehtävät muutokset voivat myös vaikuttaa automatisoituihin testitapauksiin, ja epäonnistuneita testiajoja pitää analysoida ja testitapauksia korjata. Testiautomaation kustannukset painottuvat projektin alkuun ja hyödyt materialisoituvat usein vasta pidemmällä aikavälillä, esimerkiksi testaajilta vapautuneen ajan sekä regressiotestien paremman kattavuuden ja ennaltaehkäistyjen virheiden myötä. (Guberna 2020.)

Testiautomaatio on olennainen osa jatkuvan integraation ja toimituksen putkea, josta puhuttaessa yleisesti käytetään lyhennettä CI/CD. Tällä tarkoitetaan jatkuvien koodimuutosten, uusien koontiversioiden ja niiden testauksen sekä uusien koontiversioiden käyttöönoton automaattista prosessia. Malli on osa ohjelmistokehityksen ketteriä menetelmiä mahdollistaen pienien sovelluspäivitysten toimituksen usein ja luotettavasti. Testiautomaatiolla on ratkaiseva rooli tämän mallin toimintakyvyssä, sillä jatkuvien muutosten takia sovelluksen koodi muuttuu jatkuvasti ja ilman automatisoitua testausta riski virheiden valumiselle huomaamatta ensin testiympäristöön ja lopulta tuotantoon on hyvin suuri. Manuaalisesti koko sovelluksen testaus kauttaaltaan jatkuvasti olisi käytännössä yhtään isomman sovelluksen kyseessä ollessa mahdotonta tai ainakin hyvin aikaa vievää ja testaajan kannalta erittäin kuluttavaa ja tylsää, jolloin myös riski testausvirheille on suuri. Testiautomaation avulla sovelluksen rikkova koodi CI/CD-putkessa havaitaan ajoissa, säästetään testaajien aikaa muihin tehtäviin, esimerkiksi automatisoitavien testitapausten kirjoitukseen tai tutkivaan testaukseen ja toisaalta myös ohjelmistokehittäjät saavat palautetta koodinsa laadusta nopeammin. (IBM Cloud Education 2021; JetBrains 2021.)

5 Tutkimusmenetelmien valinta ja luotettavuus

Opinnäytetyössä käytettiin aineistonkeruumenetelminä haastatteluja, kyselyä sekä havainnointia ja aineiston analyysimenetelmänä laadullista sisällönanalyysia. Ensimmäiset tutkimustulokset saatiin keväällä 2021 järjestetystä Valan omasta workshopista, josta jatkoin syksyllä 2021 kahdella kollegan teemahaastattelulla ja testiautomaatiokehittämisen havainnointina asiakasprojektissa.

Opinnäytetyötä suunnitellessani ja vielä sen kuluessa pohdin sitä, mitkä menetelmät tarjoaisivat minulle parhaat edellytykset ymmärtää, millaisia askeleita minun ja muiden samassa tilanteessa olevien tulisi ottaa testiautomaation suhteen. Koko ajan päällimmäisenä ajatuksena oli käyttää ensisijaisesti haastatteluja syvällisemmän tiedon hankintaan, sillä anonyymeissä

lomakekyselyissä vastaukset tyypillisesti jäävät pinnallisemmiksi kuin haastatteluissa ja tarpeeksi tarkkoja kysymyksiä vastausvaihtoehtoineen voi olla hankala muodostaa aiheesta, josta ei ole omaa käytännön kokemusta. Pohdin aluksi myös ohjelmistotestaajille suunnatun laajemman kyselyn tekemistä, mutta Valan workshopin ja Valan testiautomaatiotyökalukyselyn tulosten näkemisen jälkeen en nähnyt sitä enää tarpeellisena. Workshopin kyselyssä saatiin kysymyksiin kymmenestä kahdeksaan vastaajaa kysymystä kohden tutkimuksen kannalta oikeanlaiselta kohderyhmältä ja 2020 joulukuussa toteutetussa Valan testiautomaatiotyökalukyselyssä vastaajia oli vielä tuplasti enemmän, joten en kokenut enää tarpeellisena tutkimuksen reliabiliteetinkaan kannalta lähteä tekemään laajempaa kyselyä, varsinkin kun tutkimus on luonteeltaan laadullinen tutkimus, jossa tutkimusaineiston laadulla on enemmän merkitystä kuin sen absoluuttisella määrällä. Koin että riski sille, että tutkimuksen tulokset olisivat harhaanjohtavia Suomen mittakaavassa, on varsin pieni ja toisaalta sain mielestäni hyvin vastauksia tutkimuskysymyksiin valituilla menetelmillä.

Apuna haastattelukysymysten laatimiseen minulla oli Valan esimieheni, jolla on pitkä kokemus testiautomaatiosta ja joka oli myös toinen teemahaastateltavistani. Teemahaastatteluihin päädyin perehdyttyäni kattavasti erilaisiin haastattelutyyppeihin. Halusin kuulla haastateltavien kertovan omin sanoin omista kokemuksistaan keskusteltavista teemoista ja koin, että liian strukturoitu muoto voisi jättää jotain olennaista tietoa pois, koska en osaisi omalla kokemuksellani kysyä kaikkia oikeita kysymyksiä oikeine vastausvaihtoehtoineen. Kun itselläni ei ole aiheesta omaa kokemusta tai syvällistä tietopohjaa, niin avoin tai syvähaastattelu olisi myöskin ollut vaikea toteuttaa.

Tutkimuksen validiteetin osalta voin sanoa, että määrällisen tutkimuksen mukainen laajempi kysely olisi voinut mahdollisesti tuoda lisää tietoa erilaisista käytettävistä testiautomaatiotyökaluista, mutta välttämättä se ei olisi juurikaan tuonut tutkimuskysymysten suhteen lisäarvoa, sillä ensisijaiset testiautomaation työkalut Suomen markkinoilla etenkin aloitteleville testiautomaatiokehittäjille näyttävät tämän tutkimusaineiston ja Valan testiautomaatiotyökalukyselyn perusteella aika selviltä.

5.1 Vala Automation Center of Excellence Workshop

Keväällä 2021 pidettiin Valalla sisäinen workshop opinnäytetyötä vastaavasta aiheesta, eli ohjelmistotestaajien poluista testiautomaatiokehittäjäksi. Workshopin avasi kahden testaajakollegan esitykset heidän omista poluistaan testiautomaatiokehittäjiksi. Näiden esitysten jälkeen osallistujat ryhmähaastattelumaisesti vuorotellen kertoivat omasta testiautomaatiopolustaan samoihin tukikysymyksiin nojaten. Lopuksi osallistujat vastasivat vielä anonyymisti online-kyselyyn, jonka kysymysten laatimiseen olin osallistunut.

5.1.1 Esitykset

Ensimmäinen esittäjä oli aloittanut uransa ohjelmoijana monia vuosia sitten, tehnyt välillä ihan muillakin aloilla sekä ulkomailla töitä, mutta palasi lopulta takaisin IT-alalle ja Suomeen testauksen pariin useita vuosia sitten. Hän on kuluneina vuosina ollut useissa erilaisissa testausprojekteissa ja edellisellä työnantajalla oli päässyt tutustumaan hieman myös testiautomaatioon, mutta ei päässyt kuitenkaan siihen syvällisesti perehtymään. Valalla onneksi testiautomaatioon on ollut mahdollista päästä sisään, yritys tukee testiautomaatiota ja sen kehitystä vahvasti asiakasprojekteissaan. Esittäjän omassa asiakasprojektissa testiautomaatiokehittämisen aloittaminen vaati aika paljon asiakkaan vakuuttelua, mutta lopulta asiakaskin halusi lähteä rakentamaan testiautomaatiota heille.

Hän aloitti Valalla testiautomaatioon perehtymisen Valan omalla Robot Framework -kurssilla. Hän on suorittanut sen jälkeen Youtube-kursseja, etsinyt lisätietoa Pythonista sekä Robot Frameworkista ja harjoitellut samalla itsenäisesti. Testiautomaation opettelu sivutyönä normaalin manuaalisesta testaustyön ohella on hänen mukaansa haastavaa. Testiautomaation opettelu on aikaa vievää, hidasta ja vaatii pitkäjänteisyyttä, sillä kaikkea siihen liittyvää ei voi oppia kerralla.

Toinen esittäjä aloitti testausuransa Nokialla ja siirtyi myöhemmin Valalle noin viisi vuotta sitten. Hänellä ei ollut aikaisempaa ohjelmointitaustaa, vaan hän on siirtynyt testiautomaatioon käyttöön manuaalitestaajasta. Nokialla hän käytti jo joitakin testiautomaatiotyökaluja, mukaan lukien Robot Frameworkia ja Pythonia. Ensimmäisessä Valan asiakasprojektissa hän käytti Robot Frameworkia ja Pythonia testiautomaatioon, silloin kun siihen oli aikaa manuaalitestauksen jälkeen, ja sai siihen apua toiselta kollegalta Valalla. Seuraavassa projektissa ei ollut testiautomaatiota ennestään, joten siellä hän aloitti sen kehittämisen Robot Frameworkilla ja teki testiautomaatiota aina kun siihen oli aikaa manuaalitestauksen jälkeen.

Nykyisessä asiakasprojektissaan hän tekee testiautomaatiota manuaalitestauksen ohella ja käyttää testiautomaatiotyökaluina Robot Frameworkia, Seleniumia sekä Robot Framework Browseria yhdessä Jenkinsin kanssa, jota käytetään jatkuvan julkaisun ja integraation mallissa automaattitestien ajamiseen. Projektissa on haluttu käyttää Robot Frameworkia pääasiassa ja pitää Pythonin määrä minimissä, koska uusien testaajien on helpompi oppia käyttämään Robot Frameworkia kuin Pythonia, vaikka joitakin asioita olisi itse asiassa helpompi tehdä Pythonilla.

Hän kertoi käyttävänsä testiautomaatiota, koska pitää siitä, se säästää aikaa ja auttaa regressiotestauksessa. Heidän isossa kehitystiimissään sovelluskoodiin tulee tunnissakin useita muutoksia. Testiautomaatio-osaaminen manuaalitestausosaamisen rinnalla tuo hänen itsensä lisäksi myös yritykselle arvoa, kun uusia sopivia asiakasprojekteja on sen myötä helpompi löy-

tää. Hän korosti kuitenkin joustavuutta testiautomaatiokehityksessä asiakkaan suuntaan, testiautomaation tekeminen ei saisi esimerkiksi vaarantaa uusien ominaisuuksien manuaalisen testauksen aikataulua.

Vinkkeinä testiautomaation oppimiselle hän mainitsi perus- ja edistyneen tason Robot Framework -kurssit sekä Code Academyn Python -kurssin. Hänen mukaansa parhaiten oppii tekemällä itse, mikäli mahdollista niin esimerkiksi asiakasprojektin todellisessa käyttöympäristössä. Mikäli haluaa oppia testiautomaatiokehittäjäksi, niin tulisi pyrkiä sellaiseen projektiin, jossa testiautomaatio on jo käytössä. Mikäli ei ole aiemmin tehnyt testiautomaatiota, niin siihen on helpompi päästä siihen sisään sellaisessa ympäristössä, jossa testiautomaatiota jo käytetään. Toisaalta mikäli joutuu aloittamaan testiautomaatiokehittämisen tyhjästä, niin silloin siitä saa paremman kokonaisnäkemyksen, kun joutuu tekemään kaiken itse.

5.1.2 Ryhmähaastattelu

Ryhmähaastattelussa testiautomaatiokehittäjät kertoivat yksitellen kokemuksistaan testiautomaatiokehittämisen aloittamisesta, miksi he aloittivat testiautomaation parissa, millaisilla teknologioilla ja millaista apua he saivat siihen.

Ensimmäinen haastateltava kertoi aloittaneensa entisellä työpaikallaan osallistumalla Knowitin pitämään Robot Framework -koulutukseen, jonka jälkeen hän alkoi tekemään testiautomaatiota yrityksessä. Ennen koulutusta hän ei vielä tuntenut testiautomaatiota. Yrityksessä testiautomaatio oli ollut aikaisemmin käytössä, mutta sen ylläpito oli lopetettu ja testiautomaatio oli hetkeksi kuopattu, mutta se haluttiin nyt koulutuksen myötä ottaa uudelleen käyttöön. Knowitin Robot Framework -koulutuksen lisäksi hän ei ole käynyt muita kursseja, vaan opetellut lähinnä itsenäisesti sekä Valalla ollessaan hyödyntänyt myös kollegoiden apua.

Toinen testiautomaatiokehittäjä kertoi aloittaneensa Nokialla kuvapohjaisen työkalun kanssa, jota oli hankala käyttää. He olivat projektissa kuitenkin innoissaan sen käytöstä ja kokivat että testiautomaatio on sellainen asia, joka pitää ottaa haltuun. He näkivät, että kanssa oltiin asian ytimessä ja sen olevan todella käyttökelpoinen osa testausta. Myöhemmin hän teki testiautomaatiota Robot Frameworkilla, kunnes päätyi projektiin, jossa käytettävät työkalut eivät olleet lainkaan tilanteeseen soveltuvia. Hän lopulta päätyi käyttämään, ilman ohjelmointitaustaa, samaa ohjelmointikieltä testiautomaation kehittämiseen kuin sovelluksen kehittäjät käyttivät ja oppi siinä paljon hyödyntäen projektin sovelluskehittäjien ja arkkitehdin apua. Hän korosti tiimin tuen olevan tärkeää alussa opetellessa. Olen haastatellut häntä vielä myöhemmin opinnäytetyössä syvällisemmin.

Kolmas testiautomaatiokehittäjä kertoi aloittaneensa testiautomaatiokehittämisen ohjelmointitaustalla, siirtyen ohjelmoinnista testauksen puolelle todettuaan sen mielenkiintoisemmaksi

työksi. Hän aloitti testiautomaation C++ -ohjelmointikielellä ja siirtyi jonkin ajan kuluttua Robot Frameworkiin, jossa Robot Frameworkin omat nettisivut auttoivat häntä. Valalla hän sai kollegoilta vinkkejä hyvistä Robot Framework kirjastoista.

Neljäs haastateltava kertoi ajautuneensa testiautomaation pariin työharjoittelijana Nokialla, kun hänen tehtävänään oli seurata automatisoitujen testiajojen tuloksia. Siellä hän pääsi Robot Framework koulutuksiin ja Nokialla töiden päätyttyä hän kävi Sarasen järjestämän puolen vuoden Software Academy testiautomaatiokoulutusohjelman myötä Valalle töihin. Hänellä oli Valalla mentori, jonka kanssa hän oppi Pythonia, Robot Frameworkia, testiautomaatiota ylipäätänsä ja etenkin teknisiä taitoja. Koulutusohjelman jälkeen hän jatkoi Valalla entisessä asiakasprojektissaan ja nykyään hän on toisessa asiakasprojektissa yksin kehitystiimissään tekemässä testiautomaatiota Pythonin, Robot Frameworkin, Seleniumin ja Jenkinsin voimin. Hän kertoi myös yrittävänsä saada siellä Robot Frameworkia enemmän Python-muotoiseksi, koska sillä on helpompi tehdä monimutkaisia asioita.

Viides testiautomaatio-osaaja aloitti Comiqin junior -testiautomaatiokoulutuksella ryhmän kanssa itsenäisesti opiskellen ja ryhmätehtäviä tehden. Koulutuksessa käytiin läpi Robot Framework ja Python -kirjastojen tekoa sekä jatkuvan integroinnin työkaluja kokeneempien kehittäjien kanssa. Hän ei ole kovin paljoa testiautomaatiota koulutuksen jälkeen tehnyt, joitakin kuukausia, vaikkakin asiakasprojekteissa siihen liittyviä toiveita ja tarpeita tuleekin usein vastaan. Esimerkiksi nykyisessä projektissa regressiotestaukset kestävät viikkoja ja testiautomaatiolle olisi siten selkeä tarve.

Kuudes henkilö kertoi tehneensä testiautomaatiota aina jo 90-luvun alusta lähtien ohjelmistokehittäjänä, testaajana ja systeemiasiantuntijana, erilaisissa rooleissa vuorotellen ja rinnakkain, jolloin testauksen tekeminen automaation keinoin on tuntunut luonnolliselta. Testiautomaation avulla voi tehdä robotin tekemään työn itsensä sijasta. Pääosin testiautomaatiokehys piti rakentaa itse sen aikaisilla ohjelmointikielillä. Vajaa kymmenen vuotta sitten hän tutustui Robot Frameworkiin silloisella työnantajallaan ja myöhemmin hän päätyi Valalle Valan haettua Robot Framework-osaajaa. Hänen mukaansa parhaiten oppii itse tekemällä.

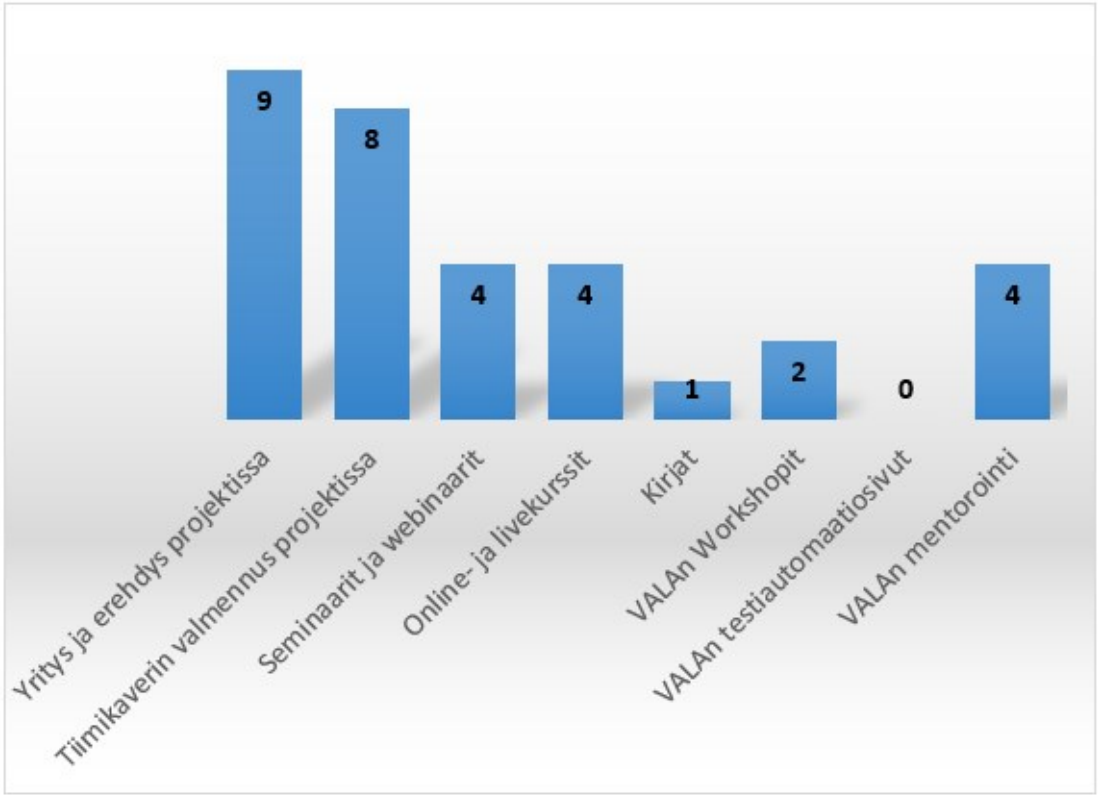
Seitsemäs testiautomaatiokehittäjä aloitti testiautomaatiouransa Nokian räätälöidyillä työkaluilla, joista siirtyi käyttämään SoapUI:ta ja sen jälkeen Robot Frameworkia ja Seleniumia. Myöhemmin hän on käyttänyt Postmania ja Newmania REST-automaation rakentamiseen Javascriptilla, JMeterilla automatisoitua kuormitustestausta Javalla, Puppeteeria ja viimeisimpänä Robot Framework Browseria Playwrightin kanssa Pythonilla täydentäen. Hän on käynyt ainoastaan Robot Frameworkin kehittäjän, Pekka Klärckin, vetämän testiautomaatiokoulutuksen ja on muuten oppinut virheiden kautta sekä kollegat ovat auttaneet tarvittaessa. Stack Overflow -verkkosivustoa hän käyttää myös edelleen lähes päivittäin ongelmatilanteissa.

Kahdeksas haastateltava kertoi, että aloittaessaan ohjelmistotestaajana hänen esimiehensä suositteli hänelle Robot Frameworkin opiskelua, jotta hän pääsee kiinni testiautomaatioon ja siihen, miten ja millaisessa kehyksessä se toimii. Hän lähti perehtymään asiaan ja kertoi sen olevan todella palkitsevaa etenkin silloin, kun onnistui ratkaisemaan jonkin pulman ja testiautomaatio lähti toimimaan. Robot Frameworkin lisäksi hänellä on kokemusta Visual Basic Scriptistä ja apua testiautomaatioon hän kertoi saaneensa esimiehiltä ja työkavereilta.

Kahdeksasta haastatellusta testiautomaatio-osaajasta kaikki kertoivat käyttäneensä Robot Frameworkia ja myös Python mainittiin useamman kerran Robot Frameworkin yhteydessä, useamman maininnan saivat myös testiautomaatiokoulutukset tai -kurssit. Itsenäinen opiskelu itse tekemällä ja työyhteisön tuen merkitys korostuivat myös vastauksissa.

5.1.3 Kysely

Kysely koostui viidestä kysymyksestä, joista neljä liittyi suoraan opinnäytetyön aiheeseen, ja joiden luontiin olin itsekin osallistunut. Ensimmäisenä oli monivalintakysymys, jossa kysyttiin mitkä tavat tällä hetkellä ovat hyödyllisiä testiautomaatiotaitojen parantamisessa (kuvio 8). Ensimmäisen kysymyksen vastauksissa korostui itsenäinen opiskelu yrityksen ja erehdyksen kautta sekä kollegoiden tuki. Myös seminaareja ja webinaareja, kursseja ja Valan mentorointia pidettiin hyvinä tapoina. Ehkä hieman yllättäen kirjat, Valan workshopit tai Valan omat testiautomaatiosivut eivät saaneet juuri kannatusta.



Kuvio 8: Hyödylliset tavat testiautomaatiotaitojen parantamisessa (Vala 2021)

Toisena kysymyksenä oli, miten Vala voisi paremmin tukea testaajia testiautomaation pariin, johon osallistujat saivat vastata avoimesti. Vastauksina tuli seuraavanlaisia ehdotuksia:

Auttaa löytämään sopiva projekti, jossa pääsee kokeilemaan testiautomaatiota. Käytännön valmennus projektissa hyvin tehokasta.

1. Enemmän tukea, valmennusta ja ajoittaisia katselmointeja asiantuntijoilta muista projekteista

2. Testiautomaation perusteiden valmennusta, mitä testiautomaatiolla voi ja ei voi tehdä. Testiautomaatiotapauksista esimerkkejä.

3. Selkeä opintopolku ja mentori. Aloitustestin avulla voitaisiin määritellä mistä aloitetaan.

4. Mentorin nimittäminen ohjaamaan testaajaa päivittäisissä testiautomaatiotehtävissä. Mentori voi arvioida testaajan koodia ja antaa kommentteja ja vinkkejä.

5. Osoittamalla miten testiautomaatio on hyödyllistä, kasvotusten tapahtuvaa oppia ja tukea. Työaikaa varataan tarpeeksi testiautomaatioon keskittymiseen.

6. Järjestämällä Robot Framework tai muita testiautomaatio -koulutuksia ajoittain. Varaamalla aikaa testiautomaation opiskeluun.

7. Kysymällä ja kuuntelemalla testaajien testiautomaatiotarpeita, järjestämällä kysymyksiä ja vastauksia tilaisuuksia ja esittämällä hyviä aloittelijatason kursseja.

8. Slack-kanava on hyvä, siellä testaaja voi pyytää apua ja järjestää vaikka puhelun kavereiden kanssa, jotka voivat auttaa.

Kolmantena kysyttiin, millaiset ensi askeleet auttavat testiautomaation oppimisessa ja mitä pitäisi välttää. Kysymykseen saatiin seuraavanlaisia vastauksia:

1. Kysyin ohjausta kokeneemmilta ja aloitin kokeilemalla sitä itse, enkä luovuttanut.

2. Yksinkertaisista tosielämän tehtävistä on hyvä aloittaa. Liika teoria ei ole hyvästä.

3. Ensiksi peruskurssit, sen jälkeen lyhyitä helppoja tehtäviä projektissa avustavan mentorin kanssa

4. Järjestämällä itselleni aikaa testiautomaation kanssa

5. Esimies suositteli mistä aloittaa. Yrityksen tarjoama polku testiautomaation opettelusta oikeisiin testiautomaatiotehtäviin projektissa on tärkeää.

6. Älä käytä liikaa aikaa vastauksien löytämiseen itse kun olet jumissa, vaan kysy kollegalta apua ja tee muistiinpanoja tulevaa varten.

7. Ala vain hommiin ja opettele lukemaan dokumentaatioita.

8. Varaa tarpeeksi aikaa harjoitteluun hyvällä aloittelijan kurssilla. Älä yritä oppia kaikkea kerralla vaan etene askel askeleelta.

9. Ohjelmointitausta auttoi sekä kollegat.

Viimeiseksi kysyttiin vielä, millaisia neuvoja antaisit testaajalle, joka haluaa oppia testiautomaatiota, ja tällaisia neuvoja annettiin:

1. Aloita tekemällä jotain äläkä murehdi virheistä, niitä sattuu paljon opetellessa ja se on ihan OK.

2. Ensiksi pitää tehdä jotain pientä, opettele askel askeleelta ja varaa siihen aikaa.

3. Etsi projektikontakti tai mentori, jolla on asiantuntemusta ja energiaa ohjata ja innostaa sinua. Etsi esimerkiksi Udemystä kurssi Robot Frameworkin perusteista.

4. Se on hidasta aluksi, mutta harjoittelu tekee mestarin. Älä luovuta!

5. Ala vain rohkeasti hommiin askel askeleelta.

6. Aloita pienillä askeleilla ja opettele koodia rivi kerrallaan. Älä yritä ymmärtää kaikkea kerralla.

7. Testiautomaatio on erittäin mielenkiintoista ja palkitsevaa työtä.

8. Lähes kaikkiin vastauksiin löytyy vastaus Googlella.

Viimeisessä kysymyksessä hieman toistuivat toiseksi viimeisen kysymyksen vastaukset, mutta muuten sanoma oli aika yhtenäinen sen suhteen mistä ja miten kannattaisi testiautomaation kanssa aloittaa. Rohkeasti aloittelijatason kurssien jälkeen pienistä ja helpommista testiautomaatiotehtävistä aloittamalla, yrittämällä ja kysymällä kokeneemmilta apua tarvittaessa. Yrityksen odotetaan tarjoavan testaajille selkeää opiskelupolkua esimerkiksi testiautomaatiokurssien muodossa, tukea, mentorointia ja valmennusta ja myös väylää testiautomaatiotehtäviin opiskelun jälkeen.

5.2 Haastattelut

Haastattelin kahta henkilöä teemahaastattelun keinoin opinnäytetyötä varten. Ensimmäisellä haastateltavalla on vuosien kokemus testiautomaatiosta, ja hän toimii tällä hetkellä noin kymmenen testaajakonsultin esimiesvastuullisena ja yrityksen osaamisen kehittämisen johtavana asiantuntijana. Haastattelun teemana oli testiautomaatio ja hänen oma testiautomaatiopolkunsa ja toisena teemana se, miten yrityksessä tuetaan ammatillista kehittymistä erityisesti testiautomaation suhteen.

Haastateltava aloitti kertomalla omasta historiastaan ja miten päätyi testiautomaation pariin. Hänen ensimmäisessä testausprojektissaan vuonna 2008 oli käytössä mobiilitestaustyökalu, jolla ajettiin automaattitestejä mobiililaitteilla. Pääosin testausta kuitenkin tehtiin manuaalisesti. Hänellä ja hänen testaajakollegallaan ei ollut siinä vaiheessa vielä aikaisempaa kokemusta testiautomaatiosta, joten testaus oli hänen mukaansa aikamoista räpellystä ja lähinnä sen sijaan, että se olisi ollut hyödyllistä, testiautomaatio herätti heissä enemmän ammatillista mielenkiintoa. Työkalun käyttö oli hidasta, se oli epävakaa ja tuotti kuvapohjaisena erikokoisilla laitteilla epävarmoja tuloksia. Hän kertoi kehitystiimissä olleen kuitenkin hyvä toisiaan tukeva ilmapiiri, jonka myötä oli helppo kokeilla ja tutkia uusia asioita.

Haastateltavalla oli tuon projektin jälkeen muutaman vuoden tauko varsinaisesta testiautomaatiokehittämisestä, kunnes se myöhemmin jatkui Robot Frameworkilla. Aluksi hänellä oli

hankaluuksia saada sitä toimimaan, ja se jäi pidemmäksi aikaa sivuun. Kokemattomana testiautomaatiokehittäjänä voi olla hankalaa selvittää, miksi joku juttu ei omalla koneella toimi ja toisaalta sen selvittely vaatisi paljon aikaa kollegoilta. Myöhemmin toisessa projektissa samoja ongelmia ei enää ollut ja hän pystyi tekemään testiautomaatiota muun tekemisen ohessa. Yhtenä kohtaamanaan ongelmana hän nosti esiin sen, että mistä löytää ajan testiautomaatiokehittämiselle, kun on vastuussa koko projektin testauksesta ja uusia ominaisuuksiakin tulee koko ajan testattavaksi. Ratkaisuna hän varasi itselleen kalenterista aikaa testiautomaatiokehittämiseen.

Yhdessä hänen projektissaan haluttiin toteuttaa testien automatisointia siihen sopimattomilla työkaluilla. Hän päätyi hylkäämään toimimattoman ratkaisun ja alkoi tekemään testiautomaatiota samalla ohjelmointikielellä, jota projektin ohjelmoijatkin käyttivät, hyödyntäen heidän osaamistaan itse opetellessaan ohjelmointikieltä ja automatisoituja testejä tehdessään. Hän korosti kehitystiimin tuen tärkeyttä prosessissa. Testiautomaatiokehitys oli teknisesti haastavaa ja hidasta, mutta asiakas oli tyytyväinen, kun hän ei luovuttanut vaan löysi keinon, jolla testiautomaation sai siinä tilanteessa toimimaan. Hänen mukaansa kun tekee jotain uutta, niin oikeiden ratkaisujen löytäminen vie enemmän aikaa eikä epäonnistumisia tai epätietoisuutta toimivasta ratkaisusta pidä pelästyä.

Kysyin haastateltavalta, missä määrin hänen mielestään ohjelmointia tulisi osata ennen kuin kykenee tekemään testiautomaatiokehittämistä. Hän kertoi sen riippuvan paljon projektista ja siitä, miten testiautomaatiota on tähän mennessä projektissa tehty ja millaisia projektin vaatimukset ovat. Jos menee olemassa olevaan kehitystiimiin, niin pitäisi sopeutua siellä käytössä oleviin toimintatapoihin. Mikäli projektissa on käytetty testiautomaatioon jotain ohjelmointikieltä, niin silloin kyseiseen ohjelmointikieleen kannattaa sopeutua ja käyttää sitä. Robot Framework testiautomaatiotyökalu sisältää kuitenkin paljon valmiita kirjastoja käytettäväksi, jolloin sitä hyödyntävän kehittäjän ei välttämättä tarvitse hallita mitään ohjelmointikieltä, vaan pelkästään kirjastojen avulla pääsee tosi pitkälle myös silloin, kun lähdetään tyhjästä liikkeelle. Projekteissa voidaankin katsoa, että käytetään vain Robot Frameworkia siitä syystä, että se ei vaadi ohjelmointikielten osaamista. Kysyin häneltä lisää Robot Frameworkista ja hän kertoi sen olevan suomalainen ilmainen avoimen lähdekoodin testiautomaatiotyökalu, joka on Suomessa hyvin suosittu, jota kehitetään jatkuvasti ja jolla on hyvin aktiivinen yhteisö ympärillään, jolloin sen käyttöön löytyy tukea helposti. Jotkut kehittäjät voivat kuitenkin kokea Robot Frameworkin käyttämisen kömpelöksi tai rajoittavaksi ja käyttävät mieluummin esimerkiksi Python ohjelmointikieltä. Paljon Pythonia käyttävät kehittäjät eivät välttämättä toisaalta tunne niin hyvin Robot Frameworkin tarjoamia valmiita kirjastoja. Valittavaa ohjelmointikieltä tai valmiiden kirjastojen käyttöä tärkeämpää on kuitenkin tehdä testit rakenteiltaan järkeviksi ja helposti ylläpidettäviksi. Kun rakenne on loogisessa järjestyksessä, niin testien ylläpitokin on helpompaa. Hän myös kertoi, ettei itselleen kannata asettaa mitään rajoituksia sen suhteen mitä työkaluja käyttää.

Haastateltava kertoi, että eniten hän on teknisesti kehittynyt silloin, kun hän on samassa projektissa tai organisaatiossa tehnyt testiautomaatiokehitystyötä muiden testiautomaatio-osaajien kanssa ristiin katselmoiden ja toisiaan sparraillen. Yksin tehdessä voi tulla sokeita pisteitä eikä silloin välttämättä oma osaaminen kehity, kun tekemistä voi jatkaa samalla tavoin vuodesta toiseen. Kun kysyin mistä hänen mielestään minun kannattaisi lähteä liikkeelle tilanteessa, jossa olen testaajana projektissa missä on myös testausautomaatiota käytössä, suositteli hän ehdottomasti hyppäämään siihen mukaan projektissa käytössä olevilla työkaluilla ja aloittamalla esimerkiksi jollain yksinkertaisemmalla testitapauksen päivityksellä. Pikkuhiljaa kun oppii lisää, voi ottaa monimutkaisempia tehtäviä. Kun lähtee käyttämään samoja projektin työkaluja, niin etuna on myös niihin ja omaan tekemiseen saatava tuki muilta organisaation kehittäjiltä sekä voi hyödyntää ja oppia valmiista aikaisemmista ratkaisuja ja tehdyistä testitapauksista.

Haastattelun toisena teemana oli yrityksen tuki ammatillisessa kehityksessä testiautomaatiokehittäjäksi. Haastateltava kertoi, että yrityksessä se on perustunut pitkälti mentorointiin sekä itseopiskeluun erilaisilla kursseilla. Myös trainee-ohjelmat ovat perustuneet mentorointeihin. Mentorointia suunnitellessa katsotaan, millaista testiautomaatio-osaamista projektissa vaaditaan ja kenellä mentorilla olisi vastaavaa osaamista. Mentorointi ei ole hänen mukaansa erityisen järjestelmällistä, siinä edistyminen edellyttää aktiivisuutta molemmilta osapuolilta ja mentorin ja mentoroitavan pitää sopia keskenään läpikäytävät asiat. Hyvää mentoroinnissa on se, että mentoroitavalla on tukihenkilö, jolta voi helposti kysyä apua. Yrityksen trainee-ohjelmissa aloittavia testiautomaatiokehittäjiä on otettu mentorin rinnalle asiakasprojektiin oppimaan. Tämä on ollut toimiva tapa, mutta mentorin kannalta myös todella vaativa ja aikaa vievä ratkaisu eikä ole täten helposti monistettavissa laajaan käyttöön, vaikka yrityksessä paljon testiautomaatio-osaamista onkin. Mentoroinnin lisäksi tai tueksi yrityksessä on suunnitteilla tällä hetkellä myös testiautomaatio-työkalujen opetteluun omia kursseja. Kurssit sisältäisivät tallennettuja luentoja sekä tehtäviä, joita lopuksi voitaisiin käydä mentorin kanssa lävitse. Tällä säästettäisiin mentorien aikaa, kun kaikkea ei tarvitse käydä alusta alkaen välttämättä yhdessä lävitse. Keskustelimme että tämä ratkaisu voisi toimia kaikille, ketkä ovat kiinnostuneita oppimaan lisää testiautomaatiosta, mentorointiohjelman ulkopuolellakin.

Toinen haastateltava oli testaajakollega, joka on hiljattain alkanut käyttämään Robot Frameworkia testien automatisointiin. Halusin haastatella häntä, koska hänellä on tuoreessa muistissa vielä mahdolliset haasteet prosessiin liittyen ja hänellä oli ennen testiautomaatiokehittämisen aloitusta samanlaiset lähtökohdat kuin itselläni on tällä hetkellä.

Haastattelun teemana oli testiautomaatio ja haastateltavan polku testiautomaatiokehittämiseen. Haastateltava oli aloittanut opiskelemalla Robot Frameworkia itsenäisesti nettikurssien avulla, joita hänelle suositeltiin. Tilanne oli aluksi hieman haastava, kun senhetkisessä pro-

jektissa testiautomaatiota ei ollut mahdollista itse tehdä. Nykyisessä se on kuitenkin mahdollista, hänellä on varsin vapaat kädet oman työnsä suhteen. Oppimisprosessiinsa hän olisi saanut yritykseltä mentorointia ja siitä jo sovittiinkin, mutta se ei kuitenkaan edennyt hänen omien työkiireidensä takia, hänen ollessa asiakasprojektissa kehitystiimin ainoa testaaja. Hän on edelleenkin ainoana testaajana projektissaan, mutta teki päätöksen opetella testiautomaatiota, vaikka välillä onkin kiireistä. Kehitystiimissä on useita muitakin Robot Framework -osaajia, joilta hän saa nyt kaiken tarvittavan tuen.

Haastateltavan mukaan kynnys aloittaa Robot Frameworkin käyttö on pienempi kuin alkaa ohjelmoida. Robot Frameworkin käyttö ei välttämättä edellytä ohjelmointiosaamista yksinkertaisempien testien osalta, mutta monimutkaisemmat testit voivat vaatia ohjelmointiosaamista ja Pythonin käyttöä.

Kun kysyin vielä, että mistä kannattaisi aloittaa testiautomaation opiskelu, niin hän suositteli käymiään Robot Framework -kursseja, sillä ne lähtevät ihan perusteista liikkeelle. Kehitysympäristön asentamisen jälkeen kurssien perustestejä tehden oppii hyvin ja tästä ei ole enää pitkä hyppäys siihen, että suunnittelee ja tekee itse omia testejään. Omaan Robot Framework harjoitteluun voi hyödyntää myös esimerkiksi isojen kauppaketjujen verkkosivustoja.

5.3 Havainnointi

Sovin omassa projektissani testiautomaatiokehittäjän kanssa, että seuraisin yhden testitapauksen automatisointia. Olin itse kirjoittanut automatisoitavan testitapauksen käyttämäämme tehtävienhallintatyökaluun JIRAan ja testiautomaatiokehittäjä oli minulle tuttu ennestään, joten en usko, että havainnointitilanne olisi jotenkin muuttanut tai häirinnyt automatisointityön kulkua tämän testitapauksen osalta, muuten kuin ehkä ajallisesti sitä hieman hidastanut. Kehittäjä selosti minulle samalla tehdessään testitapausta mitä, miten ja miksi hän teki asioita niin kuin teki, esitin itsekin välillä kysymyksiä, joten havainnointiani voi pitää osallisena havainnointina. Havainnointi tapahtui jaetun ruudun välityksellä ja tallennettiin havainnoitavan suostumuksella, jotta voin myöhemmin palata siihen tarvittaessa. Koin tämän havainnoinnin itselleni hyvin hyödyllisenä, sen näyttäessä käytännössä miten testitapauksia automatisoidaan nykyisessä projektissani.

Projektissa testiautomaatiota tehdään Java-pohjaista JBehave frameworkia ja käyttäytymislähtöisen kehittämisen periaatteita hyödyntäen. Testit kirjoitetaan niin, että niitä on helppo lukea ja tulkita ilman teknistä osaamistakin. Kehittäjän osalta kyseinen työkalu edellyttää Java-ohjelmointikielen hallintaa, joten siihen ei ole ilman ohjelmointitaustaa olevan aloittelevan testiautomaatiokehittäjän yhtä helppoa päästä sisään kuin Robot Frameworkiin. Kun kysyin testiautomaatiokehittäjältä, että millaisia Java- tai ohjelmointitaitoja JBehave-testiautomaatiokehitys vaatii, niin hänen mukaansa Javan ja ohjelmoinnin perusteiden pitäisi riittää. Omien sanojensa mukaan hän ei itsekään ole Javan erityisosaaja. Hän kertoi aloittaneensa

oman testiautomaatiopolkunsa Java- ja Selenium-verkkokursseilla ja hänellä kesti noin puoli vuotta siitä, kun hän aloitti JBehave- testiautomaation tekemisen projektissa siihen, että hän tunsi sen sujuvan hyvin.

6 Yhteenveto ja johtopäätökset

Tutkimus vastasi lopulta asettamiini tutkimuskysymyksiin hyvin. Koen tietäväni nyt, mistä ja miten minun kannattaisi lähteä liikkeelle testiautomaation suhteen omassa tilanteessani. Tutkimusaineiston perusteella testiautomaatiokehittäjäksi pyrkivällä ohjelmistotestaajalla on muutamia eri vaihtoehtoja riippuen hänen omasta tilanteestaan ja lähtökohdista. Ensinnäkin, mikäli testaajan omalla työpaikalla tai projektissa on jo käytössä testiautomaatiota, kannattaa sitä lähteä opiskelemaan työpaikan tai kehitystiimin tukemana siellä käytössä olevin työkaluin, etenkin jos kyseessä on Robot Framework. Ohjelmointikokemuksen puute ei estä Robot Frameworkin opiskelua ja käyttöä. Robot Framework on Suomessa hyvin suosittu ja sillä on laaja ekosysteemi, aktiivinen yhteisö, jonka tukeen voi ongelmatilanteissa turvautua. Sitä kehitetään aktiivisesti yhteisön toiveiden perusteella eteenpäin. Opinnäytetyön tutkimusaineistossa jokainen haastateltava mainitsi Robot Frameworkin ja Valan tekemässä kyselyssä Robot Framework oli tuttu jokaiselle vastaajalle ja suurimmalle osalle se oli myös mieluisin testiautomaatiotyökalu. Mikäli on tilanteessa, missä ei tiedä, mistä kannattaisi aloittaa testiautomaation suhteen, niin Robot Framework on aika hyvä vastaus.

Muut testiautomaatiotyökalut voivat vaatia ohjelmointiosaamista ja yleisesti testiautomaatiokehitys on luonteeltaan enemmän ohjelmistokehitystä kuin testausta, joten ohjelmoinnin perusteiden sekä jonkun ohjelmointikielen hallinta on suositeltavaa, esimerkiksi Javan tai Pythonin. Pythonia käytetään yleisesti Robot Frameworkin rinnalla ja sillä voi olla helpompi rakentaa monimutkaisia testejä kuin Robot Frameworkilla.

Mikäli ohjelmistotestaajan työpaikalla ei ole käytössä testiautomaatiota tällä hetkellä tai hän on työttömänä, niin silloin järkevintä lienee opiskella Robot Frameworkia esimerkiksi onlinekurssien avulla ja mahdollisesti pyrkiä eri yritysten järjestämiin testiautomaatiokoulutuksiin, hakea testiautomaatio trainee-ohjelmiin tai junior-positioihin ja sitä kautta pyrkiä testiautomaatiotehtäviin tulevaisuudessa.

Sen jälkeen, kun on päätetty mitä lähdetään harjoittelemaan, oli se Robot Framework tai joku muu projektissa käytössä oleva testiautomaatioratkaisu, kannattaa opiskeluun varata tarpeeksi aikaa ja pitkäjänteisyyttä. Opiskelu kannattaa aloittaa aloittelijatason kursseilla ja niiden jälkeen itsenäisesti harjoitella pienillä helpoilla tehtävillä, joista pikkuhiljaa kokemuksen karttuessa voi siirtyä vaativampiin tehtäviin. Mikäli on mahdollista saada testiautomaati-

oon mentorointia omalta työpaikalta, niin sitä tilaisuutta kannattaa hyödyntää. Kehitystiimissä olemassa olevaa osaamista kannattaa hyödyntää, eikä jäädä yksin ongelmien kanssa. Kollegoilta saatava tuki on erittäin tärkeää ja hyödyllistä prosessissa, yksin testiautomaation opiskelu voi olla hyvin haastavaa. Myös kokeneemmat kehittäjät turvautuvat kollegoidensa apuun tarvittaessa.

Yrityksessä, jossa testiautomaatio-osaamista halutaan tukea ja parantaa, tulisi kiinnittää huomiota testaajien tukemiseen testiautomaation opiskelussa tarjoamalla mentorointia, kollegoiden vertaistukea sekä selkeän opintopolun sopivine testiautomaatiokursseineen. Testiautomaation opetteluun tulisi voida käyttää myös työaikaa. Yrityksen tulisi pyrkiä tarjoamaan testaajalle väylä opiskelusta varsinaisiin testiautomaatiotehtäviin, esimerkiksi asiakasprojekteihin, joissa testiautomaation harjoittelu käytännössä aidoilla testitapauksilla olisi mahdollista, mikäli mahdollista niin kollegan tukemana.

Alun perin opinnäytetyön yhtenä tavoitteena oli tulosten perusteella päivittää myös Valan omia testiautomaation intrasivuja, mutta jouduin sen rajallisen ajan vuoksi rajaamaan ulos opinnäytetyöstä. Samoin tarkoitukseni oli itse harjoitella testiautomaatiokehittämistä tutkimuksesta saamieni vinkkien perusteella, mutta tähänkään ei lopulta jäänyt aikaa. Opinnäytetyö antaa kuitenkin hyvät eväät tarvittaessa jatkossa palata sekä yrityksen intrasivujen päivitykseen testiautomaation osalta että omakohtaisesti aloittaa testiautomaatiokehittämiseen perehtyminen käytännön tasolla.

Lähteet

Painetut

Eskola, J.& Suoranta, J. 2014 Johdatus laadulliseen tutkimukseen. Jyväskylä: Vastapaino.

Kasurinen, J-P. 2013. Ohjelmistotestauksen käsikirja. Jyväskylä: Docendo.

Moilanen, T., Ojasalo, K. & Ritalahti J. 2015. Kehittämistyön menetelmät - Uudenlaista osaamista liiketoimintaan. 3.-4. painos. Helsinki: Sanoma Pro.

Puusa, A. & Juuti, P. (toim.). 2011. Menetelmäviidakon raivaajat – perusteita laadullisen tutkimuslähestymistavan valintaan. Vantaa: Hansaprint.

Puusa, A. & Juuti, P. (toim.). 2020. Laadullisen tutkimuksen näkökulmat ja menetelmät. Tallinna: Gaudeamus.

Vilkka, H. 2021. Tutki ja kehitä. Keuruu: PS-kustannus

Sähköiset

Guberna, D. 2020. Testiautomaatio-opas. Viitattu 20.11.2021. https://www.valagroup.com/wp-content/uploads/pdf/Testiautomaatio-opas-2020.pdf

IBM Cloud Education. 2021. What Is the CI/CD Pipeline? Viitattu 20.11.2021. https://www.ibm.com/cloud/blog/ci-cd-pipeline

JetBrains. 2021. Automated Testing for CI/CD. Viitattu 20.11.2021. https://www.jetbrains.com/teamcity/ci-cd-guide/automated-testing/

MarketsAndMarkets. 2019. Automation Testing Market. Viitattu 21.11.2021. https://www.marketsandmarkets.com/Market-Reports/automation-testing-market-113583451.html

Matney, L. Facebook stock falls as global outage continues. Viitattu 28.10.2021. https://techcrunch.com/2021/10/04/facebook-stock-falls-as-global-outage-continues/

Morris, C. Facebook's outage cost the company nearly $100 million in revenue. Viitattu 28.10.2021. https://fortune.com/2021/10/04/facebook-outage-cost-revenue-instagram-whatsapp-not-working-stock/

ISTQB 2018. Sertifioitu testaaja Perustason sertifikaattisisältö. Viitattu 21.10.2021. https://tivia.fi/fistb-testi/wp-content/uploads/sites/30/2020/12/CTFL-2018-Sertifikaattisisalto-20181010-1-Valmis.pdf

ISTQB. 2021. Certifying Software Testers Worldwide. Viitattu 21.10.2021. https://www.istqb.org/

Juselius, T. 2012. Scrum-projektinhallintamenetelmä. Opinnäytetyö. Laurea-ammattikorkeakoulu. Espoo. Viitattu 21.10.2021. https://www.theseus.fi/bitstream/handle/10024/39071/Juselius_Tomi.pdf?sequence=1&isAllowed=y

Ratilainen, T. Bugien metsästys kannattaa. Viitattu 28.10.2021. https://tivia.fi/2019/12/23/bugien-metsastys-kannattaa/

Robot Framework. 2021. Viitattu 11.11.2021. https://robotframework.org/

Julkaisemattomat

VALA Automation Center of Excellence Workshop. 28.4.2021. Vala Group Oy.

VALA Test automation Tools Survey. 17.12.2020. Vala Group Oy.

Kuviot