Opinnäytetyö (YAMK)

Ohjelmistotekniikka ja ICT

2024

Saija Kaitio

Ohjelmistotestausprosessin kehitys

Opinnäytetyö (YAMK) | Tiivistelmä

Turun ammattikorkeakoulu

Ohjelmistotekniikka ja ICT

2024 | 42 sivua

Saija Kaitio

Ohjelmistotestausprosessin kehitys

Opinnäytetyön tavoitteena oli selvittää yrityksen sisäisen ohjelmistokehityksen testausprosessin nykytila, tunnistaa sen kehityskohteet ja saada ideoita prosessin parantamiseen. Tutkimuksessa syvennyttiin ohjelmistotestaukseen yleisesti, hyödyntäen alan kirjallisuutta, internetartikkeleita ja aikaisempia tutkimuksia lähdeaineistona.

Työ toteutettiin laadullisena tutkimuksena ja aineiston keruu suoritettiin lomakehaastattelujen ja havainnoinnin avulla. Lomakehaastatteluiden avulla kerättiin tuotekehitysyksikön työntekijöiden näkemyksiä nykyisestä testausprosessista, sen kehityskohteista ja parannusideoista.

Aineiston analysoinnin jälkeen voitiin esittää konkreettisia parannusehdotuksia testausprosessin kehittämiseksi. Opinnäytetyön ulkopuolelle jäi kehitetyn testausprosessin kuvaus ja käyttöönottovaihe.

Asiasanat:

Ohjelmistotestaus, testausprosessi, automaatiotestaus

Master’s Thesis | Abstract

Turku University of Applied Sciences

Software Engineering and ICT

2024 | 42 pages

Saija Kaitio

Development of the software testing process

The objective of the thesis was to investigate the current state of the company's internal software development testing process, identify areas for improvement, and gather ideas for enhancing the process. The study analyzes software testing in general, utilizing industry literature, internet articles, and previous research as source material.

The work was conducted as a qualitative study, and data collection was carried out through questionnaire interviews and observation. The questionnaire interviews were used to gather the product development unit employees' views on the current testing process, its improvement areas, and ideas for enhancements.

After analyzing the data, concrete suggestions for improving the testing process could be presented. The description and implementation phase of the developed testing process were excluded from the scope of the thesis.

Keywords:

Software testing, testing process, automated testing

Sisältö

Liitteet

Kuvat

1 Johdanto

Tämän työn tarkoituksena on kartoittaa ohjelmistotestausprosessin nykytila, tunnistaa mahdolliset haasteet ja löytää keinot kehittää prosessia vastaamaan paremmin toimeksiantajayrityksen nykyisiä vaatimuksia.

Tämän työn toimeksiantajana on kiinteistö – ja isännöintialan ohjelmistotuottaja, joka tarjoaa asiakkailleen pilvipohjaisia ohjelmistoja kattamaan koko kiinteistöhallinnan elinkaaren. Yrityksessä on noin 180 työntekijää, joista tuotekehitysyksikössä työskentelee noin 80 henkilöä. Näistä kuusi ovat testaajia, joista työn kirjoittaja itsekin on yksi.

Yrityksen laajentuessa ja ohjelmistojen monipuolistuessa on entistä tärkeämpää pitää ohjelmistoprosessit ajan tasalla, jotta voidaan tehokkaammin vastata tuleviin asiakastarpeisiin.

Tutkimusongelmana on löytää nykyisen ohjelmistotestausprosessin kehityskohteet ja esittää konkreettisia parannusehdotuksia sen tehostamiseksi. Tutkimuskysymykset ovat seuraavat: Minkälainen ohjelmistotestausprosessi tukisi parhaiten tehokasta työskentelyä? Mitkä ovat nykyisen testausprosessin kehityskohteet? Kuinka automaatiotestien kirjoittaminen voidaan integroida paremmin osaksi ohjelmistotestausprosessia?

Työn tutkimusmenetelmäksi valikoitui laadullinen tutkimus, jossa aineistonkeruuna käytettiin lomakehaastatteluja sekä havainnointia. Lomakehaastattelujen avulla pyrittiin selvittämään testausprosessin nykytila, sen kehityskohteet sekä ideoita sen parantamiseksi.

Työn teoriaosuus alkaa luvusta 2, jossa on aluksi kuvattu ohjelmistotestauksen perusteet. Alaluvuissa on esitetty teoriaa eri ohjelmistokehitysmalleista, ohjelmistotestauksen periaatteista, testaustasoista sekä testaustekniikoista. Luvussa 3 käydään läpi pintapuoleisesti yrityksen ohjelmistokehitysprosessi ja kuvataan nykyinen testausprosessi. Luvuissa 4 ja 5 on esitelty tutkimuksen toteutus, lomakehaastattelun tulokset sekä vastauksista tehdyt johtopäätökset. Lopuksi on esitelty yhteenveto, jossa on kuvattu mitä ja miksi tutkittiin, miten

tutkimus toteutettiin, millaiset tulokset tutkimuksesta saatiin sekä mikä on tutkimuksen jatkumo.

Työn teoriapohja kattaa laajasti ohjelmistotestauksen perusteet sekä teorian eri kehitysmalleista, tasoista ja tekniikoista. Keskeisinä lähteinä on hyödynnetty ISTQB:n Certified Tester Foundation Level -sertifikaatin sisältöä. ISTQB on kansainvälinen organisaatio, joka tarjoaa laajasti tunnustettuja ohjelmistotestauksen sertifikaatteja ympäri maailmaa. (ISTQB 2024.)

Lisäksi teoriaosuudessa on hyödynnetty Bernand Homesin teosta "Fundamentals of Software" sekä John Watkinsin teosta "Testing IT: An off-the-shelf software testing process”, jotka tarjoavat kattavan kuvauksen erilaisista ohjelmistokehitysmalleista, testauksen periaatteista sekä esittelevät erilaisia testaustasoja ja -tekniikoita.

Työn aikana tutkittiin myös aiempia opinnäytetöitä, jotka tarjosivat arvokasta tietoa aiheeseen. Yksi tärkeä referenssi oli Kirsi Illikaisen laatima opinnäytetyö "Testausprosessin kehittäminen", jonka päätavoitteena oli analysoida toimeksiantajan järjestelmätestausprosessin nykytilaa ja tunnistaa kehityskohteita. Tutkimus toteutettiin laadullisena toimintatutkimuksena, jonka aineiston keruu toteutettiin haastatteluilla ja havainnoinnilla. Haastattelujen avulla kartoitettiin eri rooleissa toimivien henkilöiden näkemyksiä nykytilanteesta ja kehitystarpeista. Opinnäytetyössä kokeiltiin yhtä kehitysehdotusta ja arvioitiin sen vaikutuksia, minkä tuloksena syntyi kuvaus testausprosessista ja järjestelmätestaussuunnitelma. (Illikainen 2021, 1–2.)

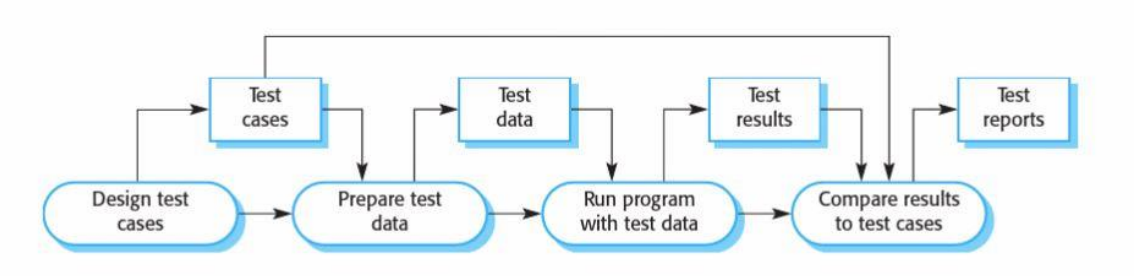
Toinen työ, jota tässä opinnäytetyössä hyödynnettiin, oli Jarmo Niemisen laatima opinnäytetyö "Testausprosessin kehittäminen – Sisäisten ohjelmistojen kehitys". Työn tavoitteena oli kartoittaa toimeksiantajan sisäisen ohjelmistokehitysprosessin nykytila ja laatia soveltuvampi testausprosessi. Aineiston keruu suoritettiin haastatteluiden ja havainnoinnin avulla. Haastatteluihin osallistui ohjelmistokehityksen parissa työskenteleviä henkilöitä, jotta saatiin kattava kokonaiskuva testausprosessista. Työssä kehitettiin päivitetty testausmenetelmä ja laadittiin prosessikuvaus, jossa kuvataan

prosessin eri vaiheet, käytettävät testausmenetelmät ja niiden vastuuhenkilöt. Lisäksi työssä luotiin dokumenttipohjat eri prosessin vaiheille. (Nieminen 2018, 9–10.)

2 Ohjelmistotestaus

Ohjelmistotestaus on osa ohjelmistokehitysprosessia, jonka avulla tunnistetaan ohjelmiston vikoja ja arvioidaan sen laatua. Sen pyrkimyksenä on varmistaa, että ohjelmisto toimii kuten se on määritelty sekä täyttää sille asetetut edellytykset. (ISTQB 2023, 15).

Yleinen erehdys on, että ohjelmistotestaus sisältää vain testien suorittamista ja tulosten tarkastelua. Ohjelmistotestaus koostuu kuitenkin myös muista toiminnoista, joista muodostuu kokonaisuudessa testausprosessi. (Kuva 1.) (ISTQB 2023, 15).



Kuva 1. Ohjelmistoprosessi malli (Sommerville 2016, 230).

Päätasolla muita toimintoja ovat testaussuunnitelman laatiminen, jossa määritellään testitavoitteet ja testauksen laajuus. Testien valvonta ja ohjaus sisältää jatkuvan seurannan kaikkien testitoimintojen edistymisestä sekä tarvittavien toimenpiteiden toteuttamisen testauksen tavoitteiden saavuttamiseksi. Testianalyysi sisältää testikohteen analysoinnin testattavien ominaisuuksien tunnistamiseksi sekä niihin liittyvien riskien määrittämiseksi ja priorisoimiseksi – se vastaa kysymykseen ”Mitä testata?”. Testisuunnitteluun sisältyy testitapausten kehittäminen vastaamaan testaussuunnitelmaa sekä testidatavaatimusten ja testiympäristön määrittely ja suunnittelu – se vastaa kysymykseen ”Miten testataan?”. Testien suorittamisen ja tulosten analysoinnin jälkeen testitoiminnot yleensä analysoidaan uusien parannusten tunnistamiseksi tulevia projekteja varten, ja luodaan testien valmistumisraportti. (ISTQB 2023, 18–19)

Testausprosessiin sisältyy ohjelmiston verifiointi sekä validointi. Verifioinnilla varmistetaan, että tuote on suunniteltu täyttämään kaikki sille asetetut toiminnalliset ja ei-toiminnalliset vaatimukset. Tämä vaihe toteutetaan yleensä kehitysprosessin alkuvaiheessa ja vastaa kysymykseen: ”Rakennanko tuotteen oikein?” (Tryqa 2023j.) Validoinnilla varmistetaan, että tuote täyttää sille asetetut vaatimukset, suorittaa sille määritellyt toiminnot ja vastaa käyttäjien sekä sidosryhmien odotuksia. Validointivaihe toteutetaan yleensä kehitysprosessin alkuvaiheessa ja sen tavoitteena on vastata kysymykseen ”Rakennanko oikeanlaisen tuotteen?” (Tryqa 2023i.)

Ohjelmistotestaus on keskeinen osa ohjelmistokehitysprosessia, sillä se auttaa varmistamaan, että sovitut tavoitteet saavutetaan aikataulussa, laadukkaasti ja budjetissa pysyen (ISTQB 2023, 16). Huolellisesti suoritettuna se estää mahdollisia vikoja, jotka voivat johtaa tarpeeseen uudelleen kehittää tuote tai pahimmillaan merkittäviin kustannuksiin ja epäonnistumiseen (Tryqa 2023h).

2.1 Ohjelmistokehitysmallit

Ohjelmistokehitysmallit ovat erilaisia prosesseja tai menetelmiä, joita käytetään ohjelmistoprojektin kehityksessä. Nämä mallit määrittelevät prosessin eri vaiheet ja niiden suoritusjärjestyksen. Kehitysmallin valinta vaikuttaa suoritettavaan testaukseen, sillä se määrittelee mitä, missä ja milloin testataan, sekä mitä testaustekniikoita käytetään. (Tryqa 2023b.)

Vesiputousmallissa vaiheet suoritetaan peräkkäin, yksi toisensa jälkeen (Homès 2012, 44). Malli on helppo ymmärtää ja käyttää, sillä jokainen vaihe on saatava päätökseen ennen kuin seuraava vaihe voidaan aloittaa. Tämän seurauksena monet viat ja puutteet havaitaan vasta myöhäisessä vaiheessa, mikä voi johtaa korkeisiin korjauskustannuksiin. (Tryqa 2023l.)

Kuten vesiputousmallissa, myös v-mallissa eri vaiheet suoritetaan peräkkäin, ja jokainen vaihe on saatava päätökseen ennen kuin seuraava vaihe voi alkaa. Tuotteen testaus suunnitellaan ja toteutetaan samanaikaisesti vastaavan kehitysvaiheen kanssa. (Tryqa 2023k.)

Spiraalimalli on yksi toistuvista kehitysmalleista, jossa jokainen sykli alkaa tavoitteiden tunnistamisella, vaihtoehtojen arvioinnilla ja rajoitusten määrittelyllä. Seuraavat vaiheet sisältävät riskien analysoinnin, tarkemman suunnittelun ja eri vaatimustasojen validoinnin vaiheittain. Lopuksi toteutetaan komponenttien koodaus, testaus, verifiointi ja hyväksyntä. (Homès 2012, 48)

Inkrementaalisen kehitysmallin periaatteena on alustava komponenttien tai toimintojen suunnittelu, johon kehitystiimi lisää toiminnallisuuksia tai lisäkomponentteja kehityksen edetessä, kunnes kehitys on valmis. (Homès 2012, 49.) Jokainen moduuli käy läpi vaatimusten, suunnittelun, toteutuksen ja testauksen vaiheet (Tryqa 2023f).

RAD-mallissa (Rapid Application Development model) komponentteja ja toimintoja kehitetään samanaikaisesti miniprojekteina, minkä jälkeen ne kootaan prototyypiksi, saadaan ensimmäinen yleiskuva ohjelmistosta ja voidaan antaa palautetta. Vaikka komponenttien nopeat muutokset ovat mahdollisia, jossain vaiheessa on tarpeen jäädyttää vaatimukset ja spesifikaatiot varmistaakseen, että komponentteja hallitaan asianmukaisesti ennen niiden toimittamista. (Homès 2012, 50.)

Ketterässä kehitysmallissa ohjelmistoa kehitetään asteittain nopeissa sykleissä, mikä johtaa pieniin inkrementaalisiin julkaisuihin. Jokainen julkaisu perustuu aiempaan toiminnallisuuteen ja testataan perusteellisesti ohjelmiston laadun varmentamiseksi. (Tryqa 2023f.)

Testaus on olennainen osa jokaista ohjelmistokehityssykliä. Sen toteutustapa vaihtelee mallista riippuen, mutta sen perusperiaatteet ovat aina voimassa (Homès 2012, 43).

2.2 Ohjelmistotestauksen periaatteet

Ohjelmistotestauksella on seitsemän pääperiaatetta, jotka muodostavat perustan ja ohjaavat testausprosessia sen eri vaiheissa.

Testaus tunnistaa vikojen olemassa olon, muta ei voi taata, ettei vikoja olisi. Ohjelmistotestaus vähentää tunnistamattomien vikojen riskiä, mutta ei voi varmistaa, että kaikki viat olisi tunnistettu.

Täydellistä testausta ei ole mahdollista saavuttaa. On tarpeen rajoittaa suunniteltujen ja suoritettavien testien määrää, jotta testaus voidaan suorittaa mahdollisimman kattavasti taloudellisten rajoitusten puitteissa. Tätä kutsutaan riskienhallinnaksi.

Testauksen aloitus varhaisessa vaiheessa perustuu siihen, että projektin kustannukset kasvavat koko sen keston ajan. On taloudellisesti järkevämpää havaita viat mahdollisimman aikaisessa vaiheessa, jotta vältytään virheellisiltä suunnitelmilta ja mahdollisilta viivästyksiltä loppuvaiheessa. Tällöin voidaan välttää virheiden kirjaaminen, korjaaminen ja uudelleen testaaminen myöhemmin, mikä edistää projektin tehokasta etenemistä.

Virheiden kasaantuminen samaan ympäristöön on yleistä, ja usein monet viat keskittyvät samoille järjestelmäkomponenteille. Tämä periaate perustuu siihen, että jos löydetään vika, voi olla tehokasta etsiä muita vikoja samasta järjestelmäkomponentista. (Homès 2012, 11–13)

Testien tehokkuus laskee, kun niitä käytetään tietty ajanjakso. Jos samoja testejä toistetaan uudelleen ja uudelleen, lopulta ne eivät enää löydä uusia vikoja. Tämän vuoksi on tärkeää tarkistaa ja kirjoittaa säännöllisesti uusia testitapauksia. (Tryqa 2023c.)

Testaus on kontekstista riippuvaista. Ei ole olemassa yhtä yleispätevää ohjetta testaukseen; se on tehtävä eri tavalla eri yhteyksissä. (ISTQB 2023, 18.)

Vikojen puuttuminen on väärinkäsitys, joka voi johtaa ohjelmiston laadun huomiotta jättämiseen. Vaikka ohjelmistosta ei löydettäisi vikoja, ei voida taata, että se vastaa vaadittuja tarpeita. Jos ohjelmisto ei vastaa käyttäjien tarpeita tai odotuksia, pelkkä vikojen tunnistaminen ja korjaaminen ei riitä. (Homès 2012, 14.)

2.3 Testaustasot

Testaustasot ovat joukko testitoimintoja, joiden avulla pyritään tunnistamaan puuttuvat alueet ja estämään toistot ja päällekkäisyydet kehitysvaiheiden välillä. Ohjelmistokehityksen elinkaarimalleissa on määritelty vaiheet, ja jokainen vaihe käy läpi testauksen. Siksi testaustasoja on useita.

Yksikkötestauksessa kehittäjät varmistavat, että heidän koodinsa toimii ja täyttää vaaditut määritykset.

Komponenttitestauksessa testataan ohjelmiston jokainen komponentti erikseen ja varmistetaan, että ne toimivat tehokkaasti.

Integraatiotestauksessa testataan integroitujen komponenttien käyttäytymistä ja toimivuutta integraation jälkeen. (Tryqa 2023a.)

Järjestelmätestaus keskittyy koko järjestelmän käyttäytymiseen ja ominaisuuksiin, mukaan lukien toiminnallisten ja ei-toiminnallisten tehtävien testaus.

Järjestelmäintegraatiotestauksessa keskitytään testattavan järjestelmän rajapintojen ja muiden järjestelmien sekä palveluiden testaamiseen. Tämä testaus vaatii yleensä samanlaisen ympäristön kuin järjestelmän lopullinen toimintaympäristö on.

Hyväksymistestauksessa keskitytään validointiin ja käyttöönottovalmiuden osoittamiseen, eli siihen, että järjestelmä täyttää käyttäjän vaatimukset ja tarpeet. Ihannetapauksessa järjestelmän aiotut käyttäjät suorittavat hyväksymistestauksen. (ISTQB 2023, 27–28.)

2.4 Testaustekniikat

Erilaisia testaustekniikoita voidaan käyttää tehostamaan testitapausten suunnittelua ja kehittämistä. Testaustekniikat on jaettu kolmeen pääryhmään, jotka esitellään tässä luvussa. (Watkins 2001, 15.)

2.4.1 Yleiset testaustekniikat

Yleiset testaustekniikat ovat korkean tason lähestymistapoja testaukseen, joita voidaan soveltaa muihin testaustekniikkaryhmiin. (Watkins 2001, 15.) Tämä ryhmä sisällään positiivisen ja negatiivisen testauksen, jotka täydentävät toisiaan. Positiivinen testaus varmistaa, että järjestelmä täyttää asetetut vaatimukset, kun taas negatiivinen testaus osoittaa, että järjestelmä ei tee sellaista, mitä sen ei pitäisi tehdä. (Watkins 2001, 16.)

Lisäksi ryhmään kuuluvat White Box- ja Black Box- testaustekniikat, jotka perustuvat tietoon järjestelmän sisäisestä rakenteesta. White Box-testit suunnitellaan ymmärtäen, miten testattava järjestelmä on rakennettu, kun taas Black Box-testauksessa testitapaukset perustuvat järjestelmän ulkoiseen käyttäytymiseen. Jos järjestelmälle on määritelty vaatimuksia, ne testataan, ja jos ei ole, voidaan perustana käyttää käyttöohjetta tai dokumentaatiota, joka kuvaa järjestelmän toimintaa. (Watkins 2001, 17.)

2.4.2 Toiminnalliset testaustekniikat

Toiminnallisilla testaustekniikoilla varmistetaan, että testattava järjestelmä täyttää sille määritellyt toiminnallisuudet (Watkins 2001, 15). Toiminnallisen testauksen voi tehdä kahdesta näkökulmasta.

1. Vaatimuspohjainen testaus: Tässä lähestymistavassa vaatimukset priorisoidaan riskikriteerien mukaan, ja testit priorisoidaan vastaavasti. Näin varmistetaan, että tärkeimmät ja kriittisimmät testit sisällytetään testaukseen.
2. Liiketoimintaprosessipohjainen testaus: Tässä näkökulmassa kuvataan skenaariot, jotka liittyvät järjestelmän päivittäiseen liiketoimintaan. Tämän testauksen pohjana toimii tieto liiketoiminnan prosessista. (Tryqa 2023e.)

2.4.3 Ei-toiminnalliset testaustekniikat

Ei-toiminnallisilla testaustekniikoilla varmistetaan, että testattava järjestelmä täyttää sille asetetut ei-toiminnalliset vaatimukset (Watkins 2001, 15). Näissä testeissä keskitytään komponentin tai järjestelmän laatuominaisuuksiin. Ei-toiminnallinen testaus kattaa ohjelmiston osat, jotka eivät liity tiettyyn toimintoon tai tehtävään. Tähän testaukseen kuuluvat muun muassa luotettavuustestaus, käytettävyystestaus, suorituskykytestaus ja kuormitustestaus. (Tryqa 2023g.)

3 Nykytila

Yrityksen tuotekehityksessä käytetään kuuden viikon kehityssprinttejä, joiden jälkeen on kahden viikon cool-down-jakso ennen uuden sprintin alkamista. Kuuden viikon sprinttien jälkeen tehdään julkaisut, joissa uudet kehitykset viedään tuotantoon asiakkaille. Välijulkaisuja tehdään vain, jos on tarvetta korjata kriittisiä virheitä. Kahden viikon cool-down-jaksolla pyritään tekemään pieniä kehityksiä, jotka parantavat järjestelmää.

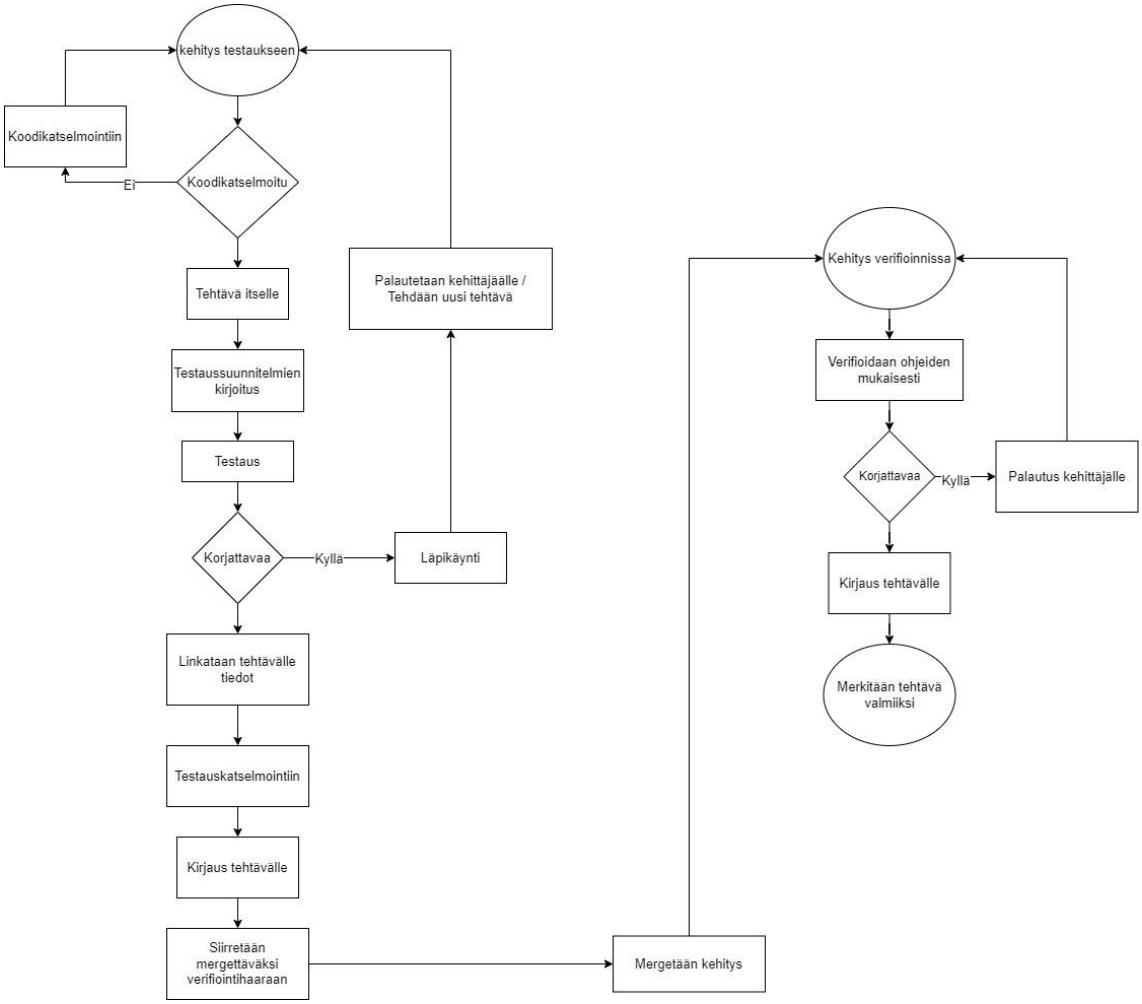
Toimeksiantajan tuotekehitysyksikkö on jaettu kahteen tiimiin, joista toinen tiimi on edelleen jaettu kahteen eri linjaan. Jokaisessa linjassa työskentelee kaksi testaajaa. Vaikka testaajat ovat erikoistuneet oman linjansa alueisiin, he voivat tarvittaessa auttaa myös muiden linjojen testauksessa.

Testaustiimiin kuuluu kuusi vakituista testaajaa, joihin opinnäytetyön tekijä kuuluu. Jokainen tiimin jäsen pyrkii tekemään manuaalitestauksen ohella automaatiotestejä robotilla sekä ylläpitämään robotin toimivuutta. Aiemmin tiimiä johti testauspäällikkö, mutta hänen jäätyä pois, tilalle ei ole nimetty uutta esimiestä.

3.1 Nykyisen testausprosessin kuvaus

Kehitysprosessi alkaa korkealla tasolla tuoteomistajien tarpeen määrittelystä. Tämän jälkeen kehitystiimit arvioivat kehityksen keston. Ennen jokaisen sprintin alkua pidetään sprint planing-kokous, jossa kehitystiimi käy läpi seuraavan sprintin tavoitteet ja sitoutuvat toteuttamaan sovitut ominaisuudet.

Kun kehittäjät ovat saaneet koodin valmiiksi ja tehneet tarvittavat yksikkötestit sekä koodikatselmoinnit, kehitys siirretään testaukselle. Tämänhetkinen yksinkertaistettu testausprosessin kulku on esitetty kuvassa 2.



Kuva 2. Testausprosessi.

Testaaja aloittaa tarkistamalla, että kehittäjä on tehnyt vaaditut dokumentaatiot uusista toiminnallisuuksista. Tämän jälkeen testaaja kirjoittaa tai täydentää olemassa olevia testaussuunnitelmia ja suorittaa vaaditut manuaalitestit. Jos kehitys vaatii muita testejä, ne suoritetaan tämän jälkeen.

Jos kehityksessä löytyy virheitä tai puutoksia, ne käydään läpi kehittäjän ja mahdollisesti tuoteomistajan kanssa, jotta niiden kriittisyys voidaan arvioida. Riippuen virheen tai puutoksen vakavuudesta, kehitys palautetaan joko kehittäjälle korjattavaksi tai luodaan uusi tehtävä, jossa virhe tai puutos korjataan.

Kun kehitys on testattu hyväksyttävästi, linkitetään havainnollistava kuva sekä testaussuunnitelma, ja kirjataan verifiointiohje testattavaan ominaisuuteen. Tämän jälkeen kehitys siirretään toiselle testaajalle testauskatselmointiin.

Jos kehitys vaatii uusia automaatiotestejä tai muutoksia olemassa oleviin testeihin, pyritään ne tekemään mahdollisimman pian testauksen jälkeen, jotta testausprosessi pysyy ajan tasalla ja mahdolliset virheet havaitaan nopeasti.

Testauskatselmoinnissa testaaja käy läpi, että testeissä on testattu kaikki tarpeellinen, testaussuunnitelmat ovat saatavilla ja ne on kirjoitettu ymmärrettävästi. Lisäksi varmistetaan, että tehtävälle on laadittu verifiointiohje, jonka perusteella kehityksen pystyy verifioimaan, ja että kehityksestä on havainnollistava kuva.

Jos tämä kaikki on kunnossa, kirjataan tehtävälle tieto siitä ja siirretään kehitys yhdistettäväksi verifiointihaaraan.

Kun kehitys on yhdistetty verifiointihaaraan, käy testaaja verifioimassa sen verifiointiympäristössä ohjeiden mukaisesti. Jos tässä vaiheessa löytyy virheitä, palautetaan kehitys kehittäjälle ja käydään virhetilanne läpi. Jos virheitä tai puutoksia ei löydy, kirjataan tehtävälle tieto verifioinnista ja siirretään tehtävä valmiiksi tilaan.

4 Tutkimus– ja kehittämismenetelmät

Tämän tutkimuksen tavoitteena on arvioida yrityksen sisäisen ohjelmistokehityksen nykytilaa testausprosessin näkökulmasta. Tutkimuksen päämääränä on tunnistaa mahdolliset pullonkaulat ja kehitystarpeet, jotka voisivat parantaa testausprosessin tehokkuutta vastaamaan tuleviin asiakasvaatimuksiin.

4.1 Tutkimusmenetelmät

Tutkimuksen päätavoitteena on selvittää, miten nykyistä testausprosessia voitaisiin kehittää tehokkaammaksi. Tehokkaammalla testausprosessilla tarkoitetaan prosessia, joka tukee paremmin tuotekehityksen tarpeita, nopeuttaa kehityssykliä ja parantaa ohjelmiston laatua. Tähän päämäärään pyrittiin hyödyntämällä laadullisen tutkimuksen menetelmiä, joiden tavoitteena on kehittää ja päivittää tutkittavaa toimintaa.

Laadullisessa tutkimuksessa keskitytään ymmärtämään tutkittavaa toimintaa, selittämään sen koostumusta, tekijöitä sekä niiden välisiä suhteita. Tämän menetelmän avulla pyritään keräämään laadullista tietoa, kuten havaintoja, haastatteluja ja dokumentteja, joilla saadaan syvällisempi ymmärrys tutkittavasta toiminnasta. (Kananen 2013, 26–27).

Aineiston keruu tutkimukseen toteutettiin lomakehaastatteluiden ja havainnoinnin avulla. Lomakehaastattelujen avulla kerättiin tuotekehitysyksikössä työskentelevien henkilöiden näkemykset nykyisestä testausprosessista, sen kehityskohteista ja ideoista prosessin parantamiseksi. Lomakehaastatteluja täydennettiin havainnoinnilla kerätyn aineiston avulla.

Laadullista analyysia voidaan tehdä monilla eri tavoilla. Analyysimenetelmä viittaa siihen konkreettiseen tapaan, jolla aineistoa käsitellään ja analysoidaan. Laadullisessa tutkimuksessa käytetään perinteisesti koodaamista, teemoittelua ja tyypittelyä analyysin välineinä. (Tietoarkisto 2021.)

4.2 Tutkimuksen toteutus

Tutkimusaineiston keruu perustui ensisijaisesti yrityksen tuotekehityksessä työskentelevien lomakehaastatteluihin sekä havainnointiin. Lomakehaastatteluilla haluttiin selvittää testausprosessin nykytila sekä sen kehityskohteet ja ideat.

Lomakehaastattelu lähetettiin tuotekehitysyksikön eri rooleissa työskenteleville, kattaen erilaisia tehtäviä ja vastuualueita, jotta saatiin monipuolinen näkemys testauksen tarpeista ja kehityskohteista.

Ennen lomakkeen lähettämistä vastaajien kanssa käytiin läpi tutkimuksen tarkoitus, tavoite ja aikataulu. Kaikille osallisille lähetettiin sama lomakehaastattelu linkki sähköpostitse ja vastausaikaa annettiin kolme viikkoa.

Haastattelu lähetettiin yhdeksälletoista asiantuntijalle tuotekehitysyksiköstä ja siihen vastasi kahdeksantoista henkilöä eri rooleista: viisi testaajaa, kahdeksan kehittäjää, kaksi ohjelmistoarkkitehtiä, kaksi tuoteomistajaa sekä tuotekehitysyksikön esimies.

Lomakehaastattelu toteutettiin Microsoft Formsilla. Lomakkeella oli kaksikymmentäyksi kysymystä neljästä eri teemasta (liite 1): testausprosessi, suunnittelu ja dokumentointi, testausmenetelmät ja automatisointi ja yleistä.

Haastatteluiden lisäksi tutkimusaineistoa täydennettiin havainnoimalla. Tutkimuksen havainnointi pohjautui itse toimintaan osallistumisella, sen seuraamisella sekä tutkimalla olemassa olevaa dokumentaatioita.

Havainnoimalla voidaan kerätä aineistoa tutkittavasta kohteesta. Siinä seurataan tutkittavan tapauksen tai kohdeilmiön toimintaa. Prosessien toiminnot on yleisesti dokumentoitu ja kuvattu mutta se ei tarkoita että toiminta tapahtuisi käytännössä niiden mukaisesti.

Etuna havainnoinnissa on olosuhteiden aitous, sillä toiminta tapahtuu sille luonnollisessa ympäristössä. (Kananen 2013, 88–89.)

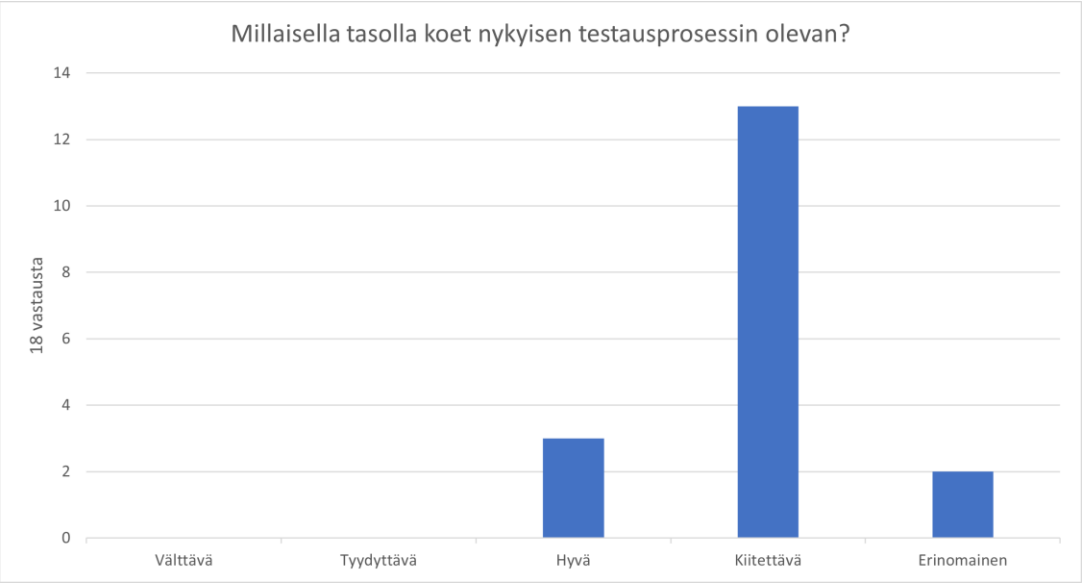
5 Tulokset

Tutkimusaineiston keruun tavoitteena oli selvittää sisäisen ohjelmistokehityksen testausprosessin nykytila sekä saada ideoita sen kehittämiseen tehokkaammaksi. Yrityksen tuotekehitysyksikössä toimiville asiantuntijoille lähetetty lomakehaastattelu sisälsi kaksikymmentäyksi kysymystä neljästä eri teemasta: testausprosessi, suunnittelu ja dokumentointi, testausmenetelmät ja automatisointi sekä yleiset kysymykset.

5.1 Testausprosessi

Lomakehaastattelu aloitettiin nykyiseen testausprosessiin liittyvällä teemalla. Kysymykset koskivat nykyisen testausprosessin tasoa, testausresursseja, testausympäristöjä, testausprosessissa muodostuvia pullonkauloja sekä parannusideoita.

Vastaajista kolme oli sitä mieltä, että nykyinen testausprosessi on hyvällä tasolla. Kolmetoista vastaajaa arvioi prosessin olevan kiitettävällä tasolla, ja kaksi piti sitä erinomaisena. (Kuva 3.)



Kuva 3. Testausprosessin nykytilan taso.

Vastauksissa korostuvat automaation ja testausresurssien riittämättömyys sekä dokumentoinnin parantamistarve. Automaatiotestauksen kehitys on hankalaa, sillä suurin osa testaajien ajasta kuluu manuaalitestaukseen, eikä testauksen kehittämiselle jää riittävästi aikaa.

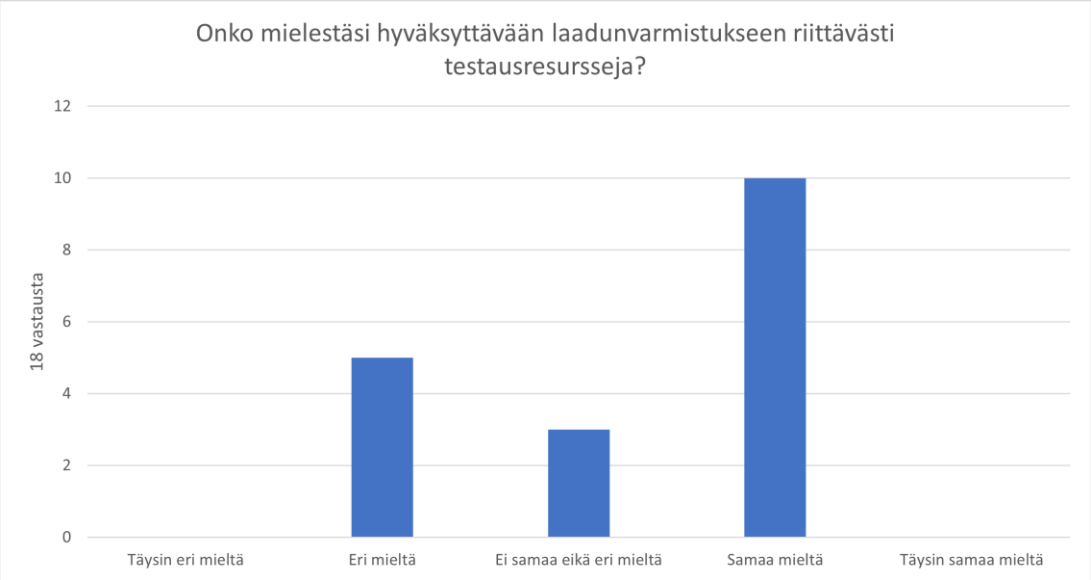
> *” Minulla on sellainen kuva että suurin osa ajasta kuluu kehittäjien työstämien yksittäisten kehitysten testaamiseen eikä varsinaiselle testauksen kehittämiselle jää aikaa. Testaajien pitäisi pystyä suunnittelemaan/ kehittämään mm. regressio testausta helpottavia asioita kehitysten testaamisen välissä. Sinänsä se työ mitä tehdään on nähdäkseni laadukasta.”*

Vastausten perusteella dokumentoinnin tulisi olla yksityiskohtaisempaa, ja testauksen suunnittelu tulisi aloittaa jo aikaisemmassa vaiheessa.

Prosessin vahvuuksina mainittiin testaajien hyvä kommunikaatio kehittäjien ja muiden testaajien välillä sekä testaustyön tarkkuus ja huolellisuus.

> *” Testausprosessi on tarkkaa ja huolellista, sekä asioista kommunikoidaan kehittäjien kanssa hyvin, mikäli asiasta on kysyttävää tai löytyy esim. uusia bugeja, että miten niiden kanssa toimitaan.”*

Lomakehaastattelussa kysyttiin, onko hyväksyttävään laadunvarmistukseen riittävästi testausresursseja. Vastaajista kaksi oli jokseenkin eri mieltä, kolme ei samaa eikä eri mieltä, ja kymmenen oli jokseenkin samaa mieltä. (Kuva 4.)

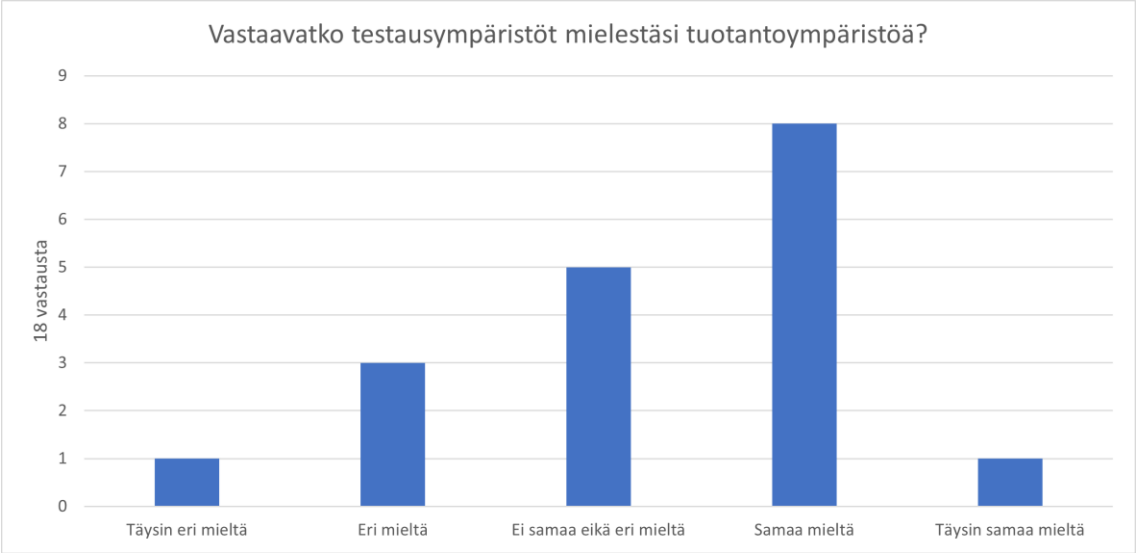


Kuva 4. Testausresurssien riittävyys.

Resurssoinnin osalta ongelmaksi nähtiin nykyisen testauksen kehityksen puute sekä automaation hidas eteneminen. Nykyisillä testausresursseilla ei jää aikaa tutustua uusiin työkaluihin tai kehittää olemassa olevia testausprosesseja.

> *” Vastasin osin tähän jo edellisessä mutta mielestäni resursseja pitäisi olla enemmän jotta testausta ehdittäisiin myös kehittää ja pystyttäisiin vähentämään manuaalitestauksen tarvetta johon se aika nykyisellään enimmäkseen kuluu.”*

Vastaajista yksi oli täysin eri mieltä siitä, että testausympäristöt vastaavat tuotantoympäristöä. Kolme vastaajaa oli jokseenkin eri mieltä, viisi ei ollut samaa eikä eri mieltä, kahdeksan oli jokseenkin samaa mieltä ja yksi täysin samaa mieltä. (Kuva 5.)



Kuva 5. Vastaavatko testausympäristöt tuotantoympäristöä.

Vastauksissa toistuu datan sekalaisuus sekä asetuskombinaatioiden haasteet. Suurten asiakasmäärien erilainen data kannassa sekä eri asetuskombinaatiot voivat aiheuttaa erilaista toimintaa järjestelmän eri osissa. Datan hallinta on haastavaa, koska testausympäristöjä käyttävät myös muut kuin testaajat.

” Periaatteessa testausympäristöt ovat melko hyvin tuotantoa vastaavia mutta kyseisiä ympäristöjä käyttää myös muut kuin testaajat ja datan hallinta ympäristöissä hankalaa”

Yksi vastaajista koki testausympäristöjen vastaavan täysin tuotantoympäristöjä.

” Teknisesti vastaa aika lailla 1:1 työntöympäristön kanssa. Jotain pieniä "epäolennaisia" asioita saattaa puuttua testausympäristöstä.”

Suurimpina pullonkauloina nykyisessä testausprosessissa nähtiin vastausten perusteella manuaalitestauksen hitaus sekä kehitysjakson loppuun sijoittuva testaus.

” Olen huomannut, että testausresurssit ovat erittäin kuormittuneita kehitysjakson loppupuolella.”

” Luonnollisesti ensin pitää koodata ja vasta sitten testata. Tämä ajoittain keskittää isoimman testaustarpeen kehityssyklin loppuun. Tätä on paikattu

testaussuunnitelmien etukäteen kirjoittamisella, testauksen aloittamisella jo kehitysvaiheessa.”

” Manuaali testauspainotteisuus on välillä pullonkaula. Myös se että uudet kehittäjät eivät välttämättä osaa kirjata oikein kaikkia asioita joita pitäisi testata.”

Automaation ja regressiotestauksen lisäys nousee eniten esille, kun kysytään nykyisen testausprosessin parantamisesta. Lisäksi vastauksissa mainitaan tarve testausympäristölle, jossa data vastaa paremmin tuotantoympäristöä.

” Automaatiotestauksen määrää pitäisi lisätä”

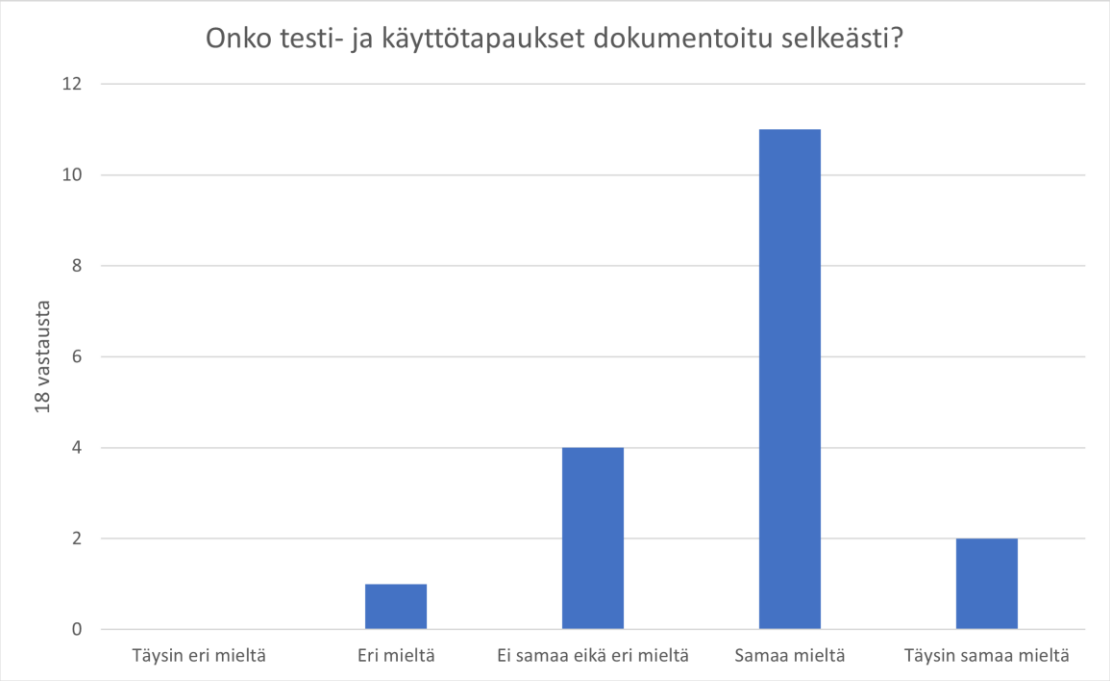
” Käytettävän datan pitäisi vastata paremmin tuotantoa.”

” Testausympäristössä voisi parantaa datan laadukkuutta/oikeanmukaisuutta, itse testausprosessissa en nää ongelmia.”

5.2 Suunnittelu ja dokumentointi

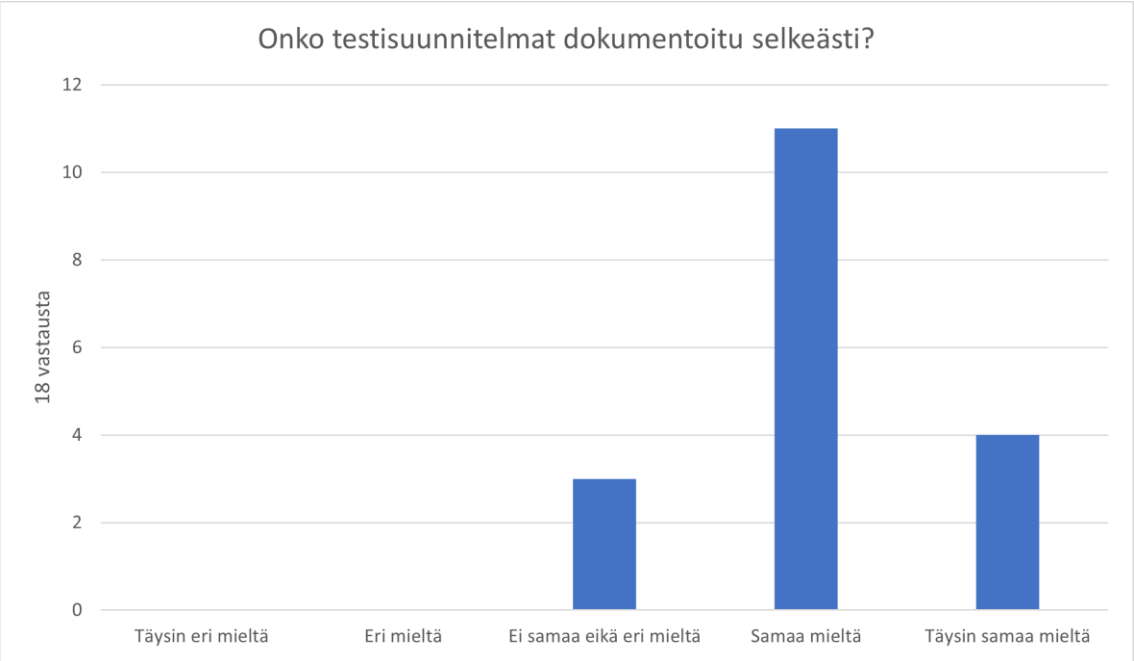
Lomakehaastattelun toinen teema koski suunnittelua ja dokumentointia. Kysymykset liittyivät testi- ja käyttötapauksien sekä testitulosten dokumentointiin sekä riskien tunnistamiseen.

Vastaajista yksi oli eri mieltä siitä, että testi- ja käyttötapaukset on dokumentoitu selkeästi, neljä vastaajaa ei ollut samaa eikä eri mieltä, yksitoista oli samaa mieltä ja kaksi vastaajaa täysin samaa mieltä. (Kuva 6.)



Kuva 6. Testi- ja käyttötapausten selkeä dokumentointi.

Kun kysyttiin testisuunnitelmien selkeästä dokumentoinnista, kolme vastaajaa ei ollut samaa eikä eri mieltä, yksitoista oli samaa mieltä ja neljä vastaajaa täysin samaa mieltä. (Kuva 7.)



Kuva 7. Testisuunnitelmien selkeä dokumentointi.

Ristiriitaisia vastauksia saatiin testi- ja käyttötapausten sekä testisuunnitelmien dokumentoinnin selkeydestä. Osassa vastaajista toivoi dokumentoinnin olevan yksityiskohtaisempaa, sisältäen esimerkkejä käytetystä testidatasta. Nykyiset suunnitelmat olettavat jo lukijalla olevan ennakkotietoa aiheesta, mikä saattaa vaikeuttaa ulkopuolisen tai uuden henkilön ymmärtämistä.

> *” Testisuunnitelmat ovat varmaankin selkeät niille henkilöille jotka niitä käyttävät työssään joskus ulkopuolisena niistä voi olla hankala saada kiinni.”*

> *” Yksityiskohtaisemmin eritoten minkälaista data on ollut per testikeissi edes esimerkin muodossa. Voi olla, että tätä on paranneltukin jo, mutta mitä nyt muistelen suunnitelmia jossain kohtaa nähneeni. Jotain yritin katsella uusia ja oli siellä jo jotain.”*

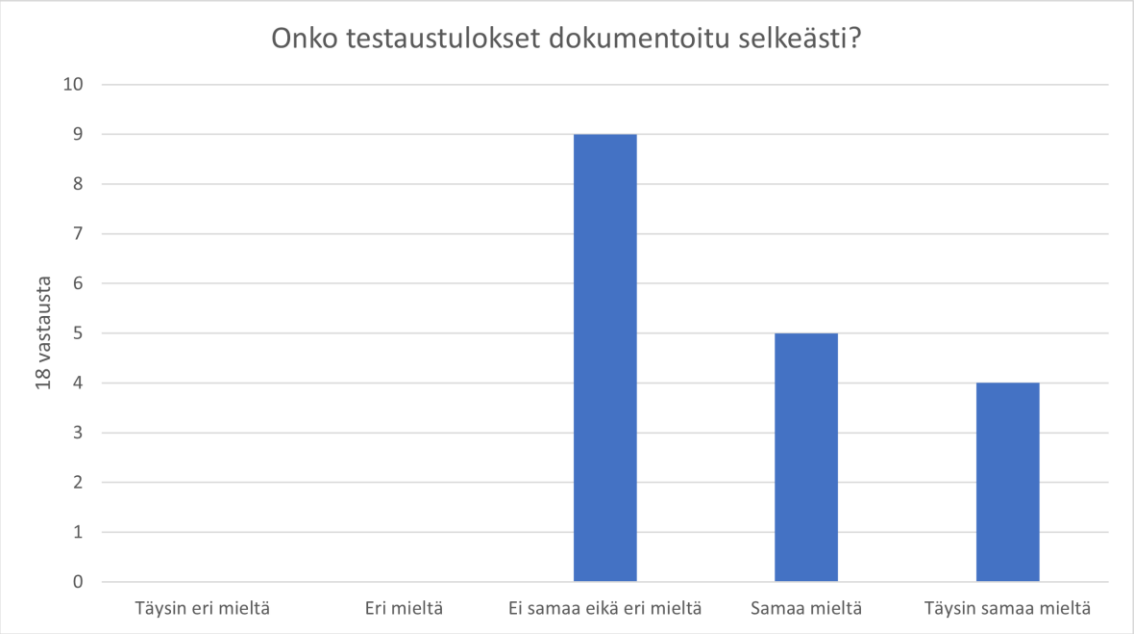
Osa vastaajista haluaisi yksinkertaistaa testisuunnitelmia, välttää tarinointia ja keskittyä vain toiminnallisuuden kuvaamiseen.

> *” Keep it simple. Vältä tarinointia (avataan asia x paikasta y, painamalla z, kannassa pitää olla tiedot a syötettynä sivulta b...). Keskity toiminnallisuuden kuvaamiseen tarinoinnin sijaan. (luo a, luo b, navigoi y, paina z, avaa x)”*

Vastauksista käy myös ilmi, että osa lomakehaastatteluun vastanneista käy vain harvoin lukemassa testausdokumentteja, eikä näin osannut ottaa kantaa niiden selkeyteen.

Kuitenkin dokumentoinnista koettiin olevan hyötyä kun esimerkiksi tarvitsee tarkistaa jonkun toiminnon toiminta tai kun kirjoitetaan kehityksen teknisiä dokumentteja.

Yhdeksän vastaajista ei ollut samaa eikä eri mieltä kun kysyttiin, että onko testaustulokset selkeästi dokumentoitu. Viisi oli samaa mieltä ja neljä vastaajaa täysin samaa mieltä. (Kuva 8.)

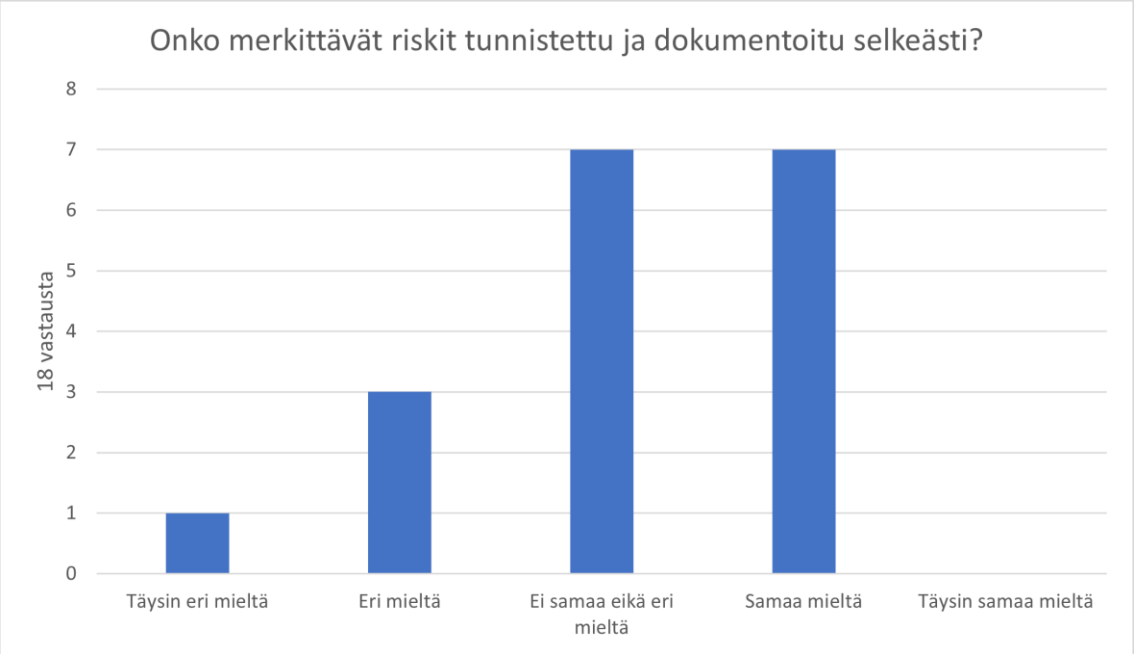


Kuva 8. Testitulosten selekä dokumentointi.

Vastauksien perusteella testituloksien dokumentoinnin paikkaa tulisi harkita uudelleen. Nykyinen käytäntö, jossa tulokset kirjataan suoraan tehtävälle, saattaa tehdä tulosten nopean selvittämisen hankalaksi. Olisi harkittava vaihtoehtoja, jotka mahdollistavat selkeämmän ja helpommin löydettävän dokumentoinnin, esimerkiksi erillisten testitulosten raportointialustan käyttöä. Lisäksi virheiden yksityiskohtaisempi kuvaaminen voisi parantaa testitulosten ymmärrettävyyttä ja auttaa kehittäjiä tehokkaammin korjaamaan havaitut ongelmat.

> *” Minulle on joskus hankala selvittää nopealla silmäyksellä, mikä testauksen tulos oli.”*

Kysyttäessä riskeistä ja niiden selkeästä dokumentoinnista, yksi vastaaja oli täysin eri mieltä, kolme eri mieltä, seitsemän ei ollut samaa eikä eri mieltä ja seitsemän oli samaa mieltä. (Kuva 9.)



Kuva 9. Riskien tunnistus ja selkeä dokumentointi.

Vastauksissa korostuu, että suurimmalla osalla vastaajista ei ole tietoa asiasta eikä ole nähnyt dokumentointia asiasta. Yhden vastaajan mielestä riskien tunnistaminen on kokemusperäistä ja dokumentoimatonta tietoa.

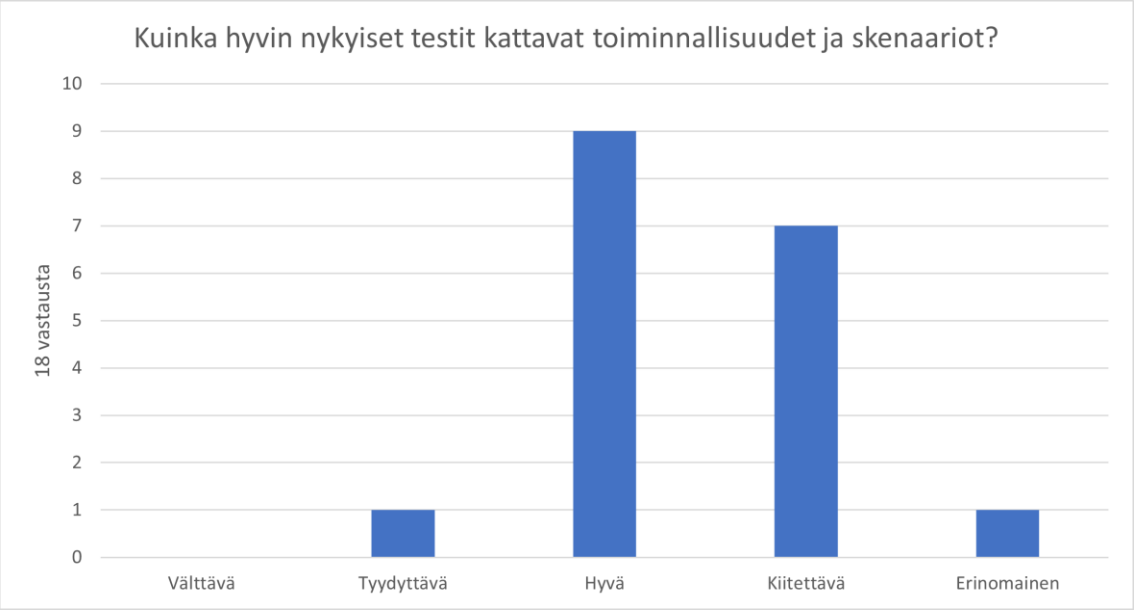
> *” Tämä on aika lailla varmaan kokemusperäistä ja dokumentoimatonta tämä tieto.”*

> *” En osaa sanoa”*

5.3 Testausmenetelmät ja automatisointi

Kolmantena teemana lomakehaastattelussa oli testausmenetelmät ja automatisointi. Kysymykset koskivat nykyisten testien kattavuutta sekä miten automaation kattavuutta saataisiin lisättyä.

Yhden vastaajan mielestä nykyiset testit kattavat toiminnallisuudet ja skenaariot tyydyttävästi, yhdeksän mielestä hyvin, seitsemän mielestä kiitettävästi ja yhden mielestä erinomaisesti. (Kuva 10.)



Kuva 10. Nykyisten testien kattavuus.

Vastaukset viittaavat siihen, että tuotteen pääasiassa käytössä olevia tai uudempia ominaisuuksia on testattu perusteellisesti. Kuitenkin vanhemmat ominaisuudet saattavat jäädä testauksen ulkopuolelle, sillä niitä ei ole päivitetty pitkään aikaan. Lisäksi monimutkaiset asetuskombinaatiot ja asiakaskohtaiset räätälöinnit tekevät kattavasta testauksesta haastavaa.

Automaatiotestien kattavuutta tulisi laajentaa. Tällä hetkellä manuaalitestaus vie suurimanosan testaajien ajasta, mikä jättää automaatiotestit kokoajan enemmän jälkeen.

” Tällä hetkellä manuaalitestaus toteutetaan kattavasti.”

” Varmasti eniten käytetyt ovat hyvin kattavia, mutta on järjestelmän osa-alueita jotka eivät ole testauksen piirissä.”

Vastauksista käy myös ilmi, että vastaajilla on epäselvää kuinka laajasti testauksessa on otettu huomioon tieturva-, kuorma- ja saavutettavuustestaus.

” Kaikkialle ei ole vielä keritty luomaan automaatiotestejä”

” Kuormitustestauksesta en ole juurikaan kuullut puhetta.”

Kysyttäessä miten automaation kattavuutta voitaisiin lisätä nykyiseen testausprosessiin, korostui vahvasti lisäresurssin tarve. Yksi ehdotus oli palkata lisää testaajia, jotka voisivat keskittyä pelkästään automaatiotestien kirjoittamiseen ja ylläpitoon. Toisen vaihtoehto oli antaa nykyisille testaajille enemmän aikaa automaatiotestauksen kehittämiseen.

Lisäksi esitettiin ideana erillisen Ad hoc -ympäristön perustamista, jotta automaatiotestejä voitaisiin ajaa ilman häiriöitä muusta työstä.

Lisäksi yksi ehdotus oli valmiiden asetuskombinaatioiden luominen robotilla, jotta niitä voitaisiin helposti ajaa testausympäristöön. Tämä voisi vähentää nykyisen manuaalisen testauksen kuormaa ja tehdä testausprosessista sujuvampaa.

” Lisää resursseja, jotta olisi aikaa tehdä niitäkin enemmän”

” Ad hoc-ympäristöt, joihin voidaan ajaa testejä ilman että se haittaa muuta työtä.”

5.4 Yleistä

Neljäntenä ja viimeisenä osiona lomakehaastattelussa oli yleisaiheinen osio. Tässä osioissa kysyttiin näkemyksiä siitä, minkä osa-alueen nykyisessä testausprosessissa koetaan tarvitsevan eniten kehittämistä. Lisäksi pyydettiin muita kommentteja tai ehdotuksia testausprosessiin liittyen.

Suurin osa vastaajista korosti automaation tarvitsevan eniten kehitystä nykyisessä testausprosessissa. Yksi vastaajista ehdotti myös testaajien osallistumista enemmän jo kehityksen suunnitteluvaiheessa.

Dokumentoinnissa kaivattiin tarkempia polkuja toiminnoille sekä testaussuunnitelmien katselmointia. Vaikka testaussuunnitelmien katselmointi onkin osa nykyisestä testausprosessia, vastausten perusteella nykyinen prosessi ei ole tarpeeksi läpinäkyvää koko tuotekehitysyksikön keskuudessa.

”Automatisointi”

” Testaajien tuloa mukaan kehityksen yhä aikaisemmassa suunnitteluvaiheessa.”

” Automatisointiin lisää resursseja eli tiimiin yksi testaaja jonka vastuaalueella on vain ja ainoastaan testausautomaation ylläpito ja kehitys.”

” Suunnittelussa, tarkoittaisi esimerkiksi suunnitelmien katselmointeja. Testauksen automatisointia tulisi lisätä (resurssipula)”

Yleisesti ottaen nykyistä testausta kehuttiin tarkaksi ja kehittäjien kanssa kommunikointi nähtiin toimivaksi. Kuitenkin vastauksissa korostettiin, että aina on mahdollista parantaa.

5.5 Johtopäätökset

Lomakehaastattelun tulosten perusteella voidaan päätellä, että suurin osa vastaajista suhtautui nykyisen testausprosessin tasoon myönteisesti. Kuitenkin selkeät kehitystarpeet ovat ilmenneet erityisesti automaation, resurssien riittävyyden ja dokumentoinnin osalta. Vaikka testaajien ja kehittäjien välinen kommunikaatio toimii hyvin, resurssien lisääminen ja suunnittelun aloituksen aikaistaminen voisivat parantaa kokonaisuutta.

Vastauksista käy selvästi ilmi automaation ja testausresurssien riittämättömyys. Suurin osa testaajista käyttää suurimman osan ajastaan manuaalitestaukseen, mikä jättää automaatiotestauksen kehittämiselle liian vähän aikaa. Lisäresurssit mahdollistaisivat automaatiotestauksen tehokkuuden ja kattavuuden parantamisen.

Tulosten perusteella testauksen suunnittelua tulisi aloittaa aikaisemmassa vaiheessa, ja testausdokumenttien selkeyttä ja tarkuutta kaivattiin. Toivottiin, että dokumentaatio olisi yksityiskohtaisempaa, jotta se olisi helpommin ymmärrettävissä ulkopuolisille ja uusille työntekijöille. Lisäksi testitulosten kirjaaminen suoraan tehtäville koettiin hankalaksi, ja niiden dokumentointipaikkaa tulisi harkita uudelleen. Vastauksissa ehdotettiin myös testaussuunnitelmien katselmointia, mikä on jo osa nykyistä prosessia. Tämä

viittaa siihen, että testausprosessi ei ole tarpeeksi läpinäkyvä koko tuotekehitysyksikölle. Yleisesti vastauksista nousee esiin vastaajien erilaiset kokemukset ja tiedot dokumentoinnista, mikä korostaa tarvetta yhtenäistää dokumenttikäytäntöjä ja varmistaa, että kaikki asianosaiset ymmärtävät niiden merkityksen ja käytön.

Vastaajista suurin osa oli jokseenkin samaa mieltä siitä, että testausympäristöt vastaavat tuotantoympäristöjä, mutta haasteina koettiin datan sekalaisuus sekä asetuskombinaatioiden monimutkaisuus. Parannusehdotuksena nousi esiin ajatus siitä, että testaukselle olisi hyödyllistä olla oma testausympäristö, joka vastaisi paremmin tuotantoympäristöä, erityisesti datan hallinnan ja asetuskombinaatioiden osalta.

Monilta vastaajilta puuttui tieto riskeistä ja niiden dokumentoinnista, mikä viittaa näiden asioiden puutteellisuuteen tai epäselvyyteen nykyisessä testausprosessissa.

Nykyisen testausprosessin vahvuutena pidettiin yleisesti testaustiimin sisäistä sekä testaajien ja koko tuotekehitysyksikön välistä kommunikaatiota. Lisäksi testaajien tarkkuutta ja huolellisuutta arvostettiin nykyisen testausprosessin vahvuuksina.

Tutkimusongelman näkökulmasta tarkasteltuna tutkimuksen tulokset osoittavat, että testausprosessin kehittämiseksi esitetyt parannusehdotukset kattavat seuraavat näkökohdat:

- Automaation lisääminen
- Resurssien lisääminen ja jakaminen
- Dokumentoinin parantaminen
- Testausprosessin suunnittelun aikaistaminen
- Testausympäristöjen parantaminen
- Riskien tunnistaminen ja dokumentointi

6 Yhteenveto

Opinnäytetyön keskittyi toimeksiantajayrityksen sisäisen ohjelmistokehityksen testausprosessin kehittämiseen. Yrityksen ja ohjelmistojen kasvaessa on entistä tärkeämpää varmistaa, että ohjelmistoprosessit ovat ajan tasalla ja työ on tehokasta.

Tutkimuksen tavoitteena oli selvittää nykyisen testausprosessin tila ja tunnistaa sen parannuskohteita, joiden avulla voitaisiin parantaa prosessin tehokkuutta ja laatua.

Tutkimusmenetelmänä käytettiin laadullista tutkimusta, joka sisälsi haastatteluja ja havainnointia. Haastatteluissa kartoitettiin tuotekehitysyksikön eri rooleissa toimivien henkilöiden näkemyksiä testausprosessin nykytilasta ja kehitystarpeista. Havainnointia käytettiin täydentämään haastatteluista saatuja tietoja ja varmistamaan tutkimuksen luotettavuus.

Tutkimuksen keskeisimmät löydökset olivat seuraavat:

- Automaation tarve: Useimmat vastaajat kokivat, että automaatiotestauksen resurssit eivät ole riittäviä. Automaatiotestauksen lisääminen ja sen kehittäminen nähtiin tärkeänä parannuskohteena.
- Testausprosessin dokumentointi: Dokumentoinnin todettiin olevan puutteellista ja epäselvää. Selkeämmät ja tarkemmat dokumentit parantaisivat prosessin läpinäkyvyyttä ja helpottaisivat uusien työntekijöiden perehdyttämistä.
- Testauksen ajoitus: Testauksen tulisi alkaa aikaisemmassa vaiheessa ohjelmistokehitysprosessia. Tämä mahdollistaisi paremman suunnittelun ja tehokkaamman testauksen.
- Yhteistyö ja kommunikaatio: Testaajien ja kehittäjien välinen kommunikaatio toimii hyvin, mutta prosessin läpinäkyvyyttä ja yhteistä suunnittelua voitaisiin parantaa.

Tutkimuksen tulosten perusteella voidaan päivittää nykyinen testausprosessi vastaamaan paremmin asiakkaiden tarpeisiin sekä tuottaa laadukkaampaa ja luotettavampaa ohjelmistoa.

Lähteet

Homès, B. 2012. Fundamentals of software testing. E-kirja Finna-palvelussa. Hoboken, New Jersey: ISTE/Wiley. Vaatii kirjautumisen palveluun. Viitattu 25.10.2023. https://ebookcentral.proquest.com/lib/turkuamk-ebooks/reader.action?docID=1120766

Illikainen K. 2021. Testausprosessin kehittäminen. Opinnäytetyö (YAMK). Tradenomi. Kajaani: Kajaanin ammattikorkeakoulu. Viitattu 18.5.2024. https://urn.fi/URN:NBN:fi:amk-202105128408

ISTQB 2023. Certified Tester. Foundation Level Syllabus. International Software Testing Qualifications Board. Viitattu 10.11.2023. https://istqb-main-web-prod.s3.amazonaws.com/media/documents/ISTQB_CTFL_Syllabus-v4.0.pdf

ISTQB 2024. About Us. Viitattu 18.05.2024. https://www.istqb.org/about-us/what-we-do

Kananen, J. 2013. Case-tutkimus opinnäytetyönä. Jyväskylän ammattikorkeakoulun julkaisuja 143. Jyväskylä: Suomen Yliopistopaino Oy.

Nieminen, J. 2018. TESTAUSPROSESSIN KEHITTÄMINEN- Sisäisten ohjelmistojen kehitys. Opinnäytetyö (YAMK). Teknologiaosaamisen johtaminen. Turku: Turun ammattikorkeakoulu. Viitattu 18.5.2024. https://urn.fi/URN:NBN:fi:amk-201803263815

Sommerville, I. 2016. Software engineering. E-kirja Finna-palvelussa. 10., uudistettu painos. Boston, Mass: Pearson. Vaatii kirjautumisen palveluun. Viitattu 19.11.2023. https://ebookcentral.proquest.com/lib/turkuamk-ebooks/reader.action?docID=5185655

Tietoarkisto 2021. Johdanto: Analyysi ja tulkinta. Laadullinen käsikirja. Viitattu 18.5.2024. https://www.fsd.tuni.fi/fi/palvelut/menetelmaopetus/kvali/analyysitavan-valinta-ja-yleiset-analyysitavat/analyysi-ja-tulkinta/

Tryqa 2023a. What are Software Testing Levels? Viitattu 12.11.2023. https://tryqa.com/what-are-software-testing-levels/

Tryqa 2023b. What are the Software Development Models? Viitattu 12.11.2023. https://tryqa.com/what-are-the-software-development-models/

Tryqa 2023c. What are the 7 principles of testing? Viitattu 25.10.2023. https://tryqa.com/what-are-the-principles-of-testing/

Tryqa 2023d. What is Agile model – advantages, disadvantages and when to use it? Viitattu 25.10.2023. https://tryqa.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/

Tryqa 2023e. What is Functional testing (Testing of functions) in software? Viitattu 18.11.2023. https://tryqa.com/what-is-functional-testing-testing-of-functions-in-software/

Tryqa 2023f. What is Incremental model- advantages, disadvantages and when to use it? Viitattu 18.11.2023 https://tryqa.com/what-is-incremental-model-advantages-disadvantages-and-when-to-use-it/

Tryqa 2023g. What is Non-functional testing (Testing of software product characteristics)? Viitattu 19.11.2023 https://tryqa.com/what-is-non-functional-testing-testing-of-software-product-characteristics/

Tryqa 2023h. What is Software Testing? Basics, Tutorial, Importance, Interview Questions. Viitattu 19.11.2023 https://tryqa.com/what-is-software-testing/

Tryqa 2023i. What is Validation in software testing? or What is software validation? Viitattu 12.11.2023. https://tryqa.com/what-is-validation-in-software-testing-or-what-is-software-validation/

Tryqa 2023j. What is Verification in software testing? or What is software verification? Viitattu 12.11.2023. https://tryqa.com/what-is-verification-in-software-testing-or-what-is-software-verification/

Tryqa 2023k. What is V-model- advantages, disadvantages and when to use it? Viitattu 12.11.2023. https://tryqa.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/

Tryqa 2023l. What is Waterfall model- Examples, advantages, disadvantages & when to use it? Viitattu 12.11.2023. https://tryqa.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/

Watkins, J. a. 2001. Testing IT: An off-the-shelf software testing process. E-kirja Finna-palvelussa. Cambridge: Cambridge University Press. Vaatii kirjautumisen palveluun. Viitattu 18.11.2023. https://ebookcentral.proquest.com/lib/turkuamk-ebooks/reader.action?docID=164738

Haastattelulomake

1. Testausprosessi
   1.1. Millaisella tasolla koet nykyisen testausprosessin olevan?
      - Välttävä
      - Tyydyttävä
      - Hyvä
      - Kiitettävä
      - Erinomainen
   1.2. Omia ajatuksia / huomioita nykyisestä testausprosessista
   1.3. Onko mielestäsi hyväksyttävään laadunvarmistukseen riittävästi testausresursseja?
      - Täysin eri mieltä
      - Jokseenkin eri mieltä
      - Ei samaa eikä eri mieltä
      - Jokseenkin samaa mieltä
      - Täysin samaa mieltä
   1.4. Omia ajatuksia testausresursseista?
   1.5. Vastaavatko testausympäristöt mielestäsi tuotantoympäristöä?
      - Täysin eri mieltä
      - Jokseenkin eri mieltä
      - Ei samaa eikä eri mieltä
      - Jokseenkin samaa mieltä
      - Täysin samaa mieltä
   1.6. Omia ajatuksia / huomioita testausympäristöstä vs. tuotantoympäristöstä?
   1.7. Onko nykyisessä testausprosessissa vaiheita, joista muodostuu pullonkauloja?
   1.8. Mitä pitäisi mielestäsi parantaa nykyisessä testausprosessissa?
2. Suunnittelu ja dokumentointi
   2.1. Onko testi- ja käyttötapaukset dokumentoitu selkeästi?

- Täysin eri mieltä
- Jokseenkin eri mieltä
- Ei samaa eikä eri mieltä
- Jokseenkin samaa mieltä
- Täysin samaa mieltä

2.2. Onko testisuunnitelmat dokumentoitu selkeästi?

- Täysin eri mieltä
- Jokseenkin eri mieltä
- Ei samaa eikä eri mieltä
- Jokseenkin samaa mieltä
- Täysin samaa mieltä

2.3. Omia ajatuksia / huomioita testitapausten ja suunnitelmien dokumentoinnista?

2.4. Onko testaustulokset dokumentoitu selkeästi?

2.5. Omia ajatuksia / huomioita testaustulosten dokumentoinnista?

2.6. Onko merkittävät riskit tunnistettu ja dokumentoitu selkeästi?

- Täysin eri mieltä
- Jokseenkin eri mieltä
- Ei samaa eikä eri mieltä
- Jokseenkin samaa mieltä
- Täysin samaa mieltä

2.7. Omia ajatuksia / huomioita riskien tunnistuksesta ja dokumentoinnista?

3. Testausmenetelmät ja automatisointi

3.1. Kuinka hyvin nykyiset testit kattavat toiminnallisuudet ja skenaariot?

- Välttävä
- Tyydyttävä
- Hyvä
- Kiitettävä
- Erinomainen

3.2. Omia ajatuksia / huomioita testien kattavuudesta?

3.3. Onko alueita tai osa-alueita, joita ei ole vielä otettu huomioon nykyisessä testausprosessissa?

3.4. Miten automaation kattavuutta saataisiin lisättyä nykyiseen testausprosessiin mielestäsi?

4. Yleistä
   4.1. Mikä testausprosessin osa-alue vaatisi mielestäsi eniten kehittämistä?
   4.2. Muita kommentteja tai ehdotuksia testausprosessiin liittyen?