LAB University of Applied Sciences

Analytiikan kehittäminen työpöytäsovellusten käyttöliittymän- ja kokemuksen tutkimista varten

LAB-ammattikorkeakoulu
Insinööri (ylempi AMK), IoT:stä tekoälyyn
2024
Juhani Nikulainen

Tiivistelmä

Tekijä(t)	Julkaisun laji	Valmistumisaika
Juhani Nikulainen	Opinnäytetyö, YAMK	2024
	Sivumäärä	
	60	

Työn nimi

Analytiikan kehittäminen työpöytäsovellusten käyttöliittymän- ja kokemuksen tutkimista varten

Tutkinto ja koulutusala

Insinööri (ylempi AMK), IoT:stä tekoälyyn

Toimeksiantajaorganisaatio (jos opinnäytetyöllä on toimeksiantaja)

Patria

Tiivistelmä

Opinnäytetyössä pohdittiin sovelluksen käyttökokemuksen merkitystä toimeksiantajan tarpeita silmällä pitäen. Määrällisen tutkimuksen keinojen joukosta etsittiin tukea laadulliselle käyttöliittymän ja käyttökokemuksen tutkimukselle. Lisäksi kartoitettiin hyödynnettävissä olevia käyttöanalytiikan menetelmiä ja työkaluohjelmistojen markkinoita. Käyttökelpoisimmaksi tuotteeksi osoittautui toimeksiantajan ohjelmistoympäristöä tukeva analytiikkajärjestelmä Qt Insight, jossa on kuitenkin ratkaisevia rajoitteita ja kehitystarpeita.

Pohdintojen ja kartoitusten perusteella nähtiin tarve uudenlaiselle analytiikkaympäristölle ja uusille analyysimenetelmille. Työssä kehitettiin analytiikkajärjestelmä, joka soveltaa yhteisoppimista ja täydentää Qt-pohjaisten sovellusten käyttöliittymän ja -kokemuksen tutkimusmenetelmiä internetistä ja pilvipalveluista riippumattomaan, tietosuojaa korostavaan suuntaan. Järjestelmä tarjoaa uusia sovellettavia analytiikkamenetelmiä ja luo pohjan paitsi laajemmin tekoälyä soveltavalle analytiikkatoiminnallisuudelle, myös esimerkiksi älykkäille käyttäjän avustamis- ja opastustoiminnoille.

Asiasanat

Käyttöliittymä, käyttökokemus, tapahtumapohjainen, analytiikka, erillisverkko, tietoturva

Abstract

Author(s)	Type of Publication	Published
Juhani Nikulainen	Master's thesis	2024
	Number of Pages	
	60	

Title of Publication

Development of analytics for studying the user interface and experience of desktop applications

Degree, Field of Study

Master of Engineering, From IoT to AI

Organisation of the client (if the thesis work is commissioned by another party)

Patria

Abstract

The thesis considered the importance of the user experience of the application, keeping in mind the employer's needs. Among the means of quantitative research, support was sought for a qualitative study of the user interface and user experience. In addition, usable usage analytics methods and tool software markets were explored. The most usable product turned out to be Qt Insight, an analytics system that supports the client's software environment, but however, has crucial limitations and development needs.

Based on the reflections and surveys, the need for a new kind of analytics environment and new analysis methods was seen. The work developed an analytics system that applies federated learning and complements research methods for the user interface and experience of Qt-based applications in a direction that is independent of the internet and cloud services and emphasizes data protection. The system offers new applicable analytics methods and a basis not only for analytics functionality that applies artificial intelligence more widely, but also for intelligent user assistance and guidance functions.

Keywords

User interface, UX, event based, analytics, private network, information security

Sisällys

1 Johdanto ... 3
1.1 Toimeksiantajan esittely ... 3
1.2 Työn tausta ja tavoitteet ... 3
1.3 Tutkimuskysymykset ja rajaus ... 4
1.4 Tutkimusmenetelmät ... 5
2 Käyttökokemuksen merkitys ... 7
2.1 Markkinoinnissa ... 7
2.2 Loppukäyttäjän arjessa ... 7
2.3 Tuotekehittäjien työlle ... 8
3 Käyttöliittymäanalytiikka osana tuotekehitysprosessia ... 9
3.1 Laadullisen tutkimuksen tukena ... 9
3.2 Tuotekehitysprosessia ohjaavana tekijänä ... 9
3.3 Turvakriittisen ympäristön haasteet ... 10
3.4 Eettisten kysymysten huomiointi ... 11
3.5 Projektointi ja jalkauttaminen ... 11
4 Käyttöliittymätutkimuksen määrällisistä tutkimusmenetelmistä ... 13
4.1 Määrälliset menetelmät laadullisten menetelmien tukena ... 13
4.2 Käyttötapaukset ja käyttäjäpolut ... 14
4.3 Käyttäjäpolkujen tunnistaminen automaattisesti ... 15
4.4 Käyttöliittymän suorituskyvyn mittaamisesta ... 15
4.5 Käyttäjäpolun kesto ... 17
4.6 Hiirellä ja näppäimistöllä tehdyn työn määrä ... 17
4.7 Tehtävien onnistuminen ... 18
4.8 Lämpökartat ... 19
5 Käyttöliittymätutkimuksen tunnetuimmat työkaluohjelmistot ... 20
5.1 Yleisimmät toiminnot ... 20
5.2 Tekniset ominaisuudet ja rajoitteet ... 20
6 Työssä käytetyt ja kehitetyt analyysimenetelmät ... 22
6.1 Yleistä ... 22
6.2 Käyttäjäpolun askelten kestot ... 22
6.3 Hiiren osoittimen kulkema skaalattu matka ... 23
6.4 Hiiren osoittimen kulkeman matkan hyötysuhde ... 24
6.5 Poikkeamien tunnistaminen käyttäjäpolkujen askelissa ... 25
6.6 Käyttäjäpolun tilastollisten tunnuslukujen käsittely ... 26
6.7 Käyttäjäpolkujen automaattinen tunnistaminen ... 27

6.8 Lämpökarttojen yhdistäminen 28
6.9 Käyttäjäpoluista laskettavat metriikat 28
6.10 Käyttäjäpolkujen yhdistäminen 29
7 Käyttöanalytiikkajärjestelmä 30
7.1 Yleistä 30
7.2 Tapahtumadata ja sen käyttäminen 33
7.3 Käyttöliittymäsovelluksen analytiikkalisäosa 35
7.4 Käyttöliittymäsovelluksen analytiikkalisäosan toimintatilat 37
7.5 Analytiikkapalvelin 40
7.6 Käyttäjäpolkujen tietorakenne 41
7.7 Tietokannat 42
7.8 Konfigurointi ja asetukset 44
8 Tulosten visualisointi ja demosovellus 45
8.1 Yleistä 45
8.2 Lämpökartat 45
8.3 Käyttäjäpolut 47
9 Yhteenveto ja pohdinta 51
9.1 Yleistä 51
9.2 Mahdollisia jatkotutkimus- ja kehitysaiheita 53
Lähteet 58

1 Johdanto

1.1 Toimeksiantajan esittely

Työ tehdään pääosin itsenäisesti, mutta työn tuloksia voidaan käyttää hyväksi Patrian Systems & Integrations -yksikössä, jos se nähdään sopivaksi asiakasprojekteissa. Ensimmäinen tutkimuskohde on mahdollisesti erään reaaliaikaisen signaalianalyysiohjelmiston interaktiivinen käyttöliittymä. Signaaliympäristön visualisoinnin lisäksi kyseinen järjestelmä tuottaa numeerista signaaliparametritietoa, jota hyödynnetään esimerkiksi tilannekuvakartan luomisessa.

Toimeksiantaja on varannut käyttöliittymän käytettävyyden tutkimustyökalujen kehittämiseen 300 tuntia työaikaa, joka käytetään analytiikkajärjestelmän ja demosovelluksen kehittämiseen.

> *”Patria on kansainvälinen puolustus-, turvallisuus- ja ilmailualan luotettu elinkaaren tukipalvelujen, lentokoulutuksen ja teknologiaratkaisujen tuottaja. Patria tarjoaa kaluston käytettävyyttä ja suorituskyvyn jatkuvaa kehittämistä sekä tiedustelu-, valvonta- ja johtamisjärjestelmien tuotteita ja palveluita ilmailu- ja puolustusalan toimijoille. Patria-konserni koostuu emoyhtiöstä, Patria Oyj:stä ja sen täysin omistamista tytäryhtiöistä. Patrian omistavat Suomen valtio (50,1 %) ja norjalainen Kongsberg Defence & Aerospace AS (49,9 %).” (Patria Group, 2024.)*

Olemme sopineet alustavasti myös Qt Companyn edustajien kanssa mahdollisesta yhteistyöstä tekoälyn tuomisessa osaksi Insight-tuotteen käyttöanalytiikkaa. Qt Insight -tuotekehitystiimi on lupautunut tukemaan analytiikan kehittämisessä ja avaamaan tuotteen sisäisiä rajapintoja opinnäytetyön tekemisen ajaksi.

> *Qt Group on maailmanlaajuinen ohjelmistoyritys, johon sekä alan johtajat että yli 1,5 miljoonaa kehittäjää maailmanlaajuisesti luottavat luodessaan käyttäjien pitämiä sovelluksia ja älylaitteita. Qt Group auttaa asiakkaitamme lisäämään tuottavuutta koko tuotekehityksen elinkaaren ajan – käyttöliittymäsuunnittelusta ja ohjelmistokehityksestä laadunhallintaan ja käyttöönottoon. (Qt Group, 2024a.)*

1.2 Työn tausta ja tavoitteet

Tuotekehitykseen kuuluu oleellisena osana käyttäjien palautteen ja toiveiden huomiointi (Balkhi 2023). Tuotekehitystiimissämme on kuitenkin tunnistettu puutteita asiakaspalautteen kantautumisessa ja hyödyntämisessä. Asiakkaiden käyttötavoista ja -tapauksista ei ole aina saatu riittävän tarkkaa tietoa ja saatu palaute on useimmiten koskenut yksittäisiä

ongelmatilanteita. Saatu palaute on voinut olla ristiriidassa toisen käyttäjäryhmän tai -roolin edustajan antaman palautteen kanssa, jolloin on tehty osaoptimointia eikä yhteistä syytä ole korjattu. Loppukäyttäjien todellinen mielipide ei välttämättä kantaudu systemaattisesti tuotteen kehittäjille. Asiakas saattaa sietää ja sopeutua tehottomiin ja hankaliinkin työnkulkuihin, jolloin optimaalinen käytön tehokkuus ja käyttäjätyytyväisyys jäävät saavuttamatta.

Signaalianalyysiohjelmiston tuotekehityksessä on tähän asti tehty käytännössä laadullista käyttöliittymän ja -kokemuksen (UI/UX) tutkimusta, jossa on keskitytty ymmärtämään käyttäjää ja hänen tarpeitaan ilman numeerista tietoa. Tavoitteena olisi kuitenkin tehdä käyttöliittymän ja -kokemuksen tutkimisesta ja kehittämisestä kattavampaa ja systemaattisempaa, ja tuoda se osaksi jatkuvaa tuotekehitysprosessia. Käyttöliittymäsuunnittelun tavoite on luoda käyttöliittymän avulla käyttäjille helppo, tehokas ja miellyttävä vuorovaikutus tuotteen kanssa (Adobe 2022a).

1.3 Tutkimuskysymykset ja rajaus

Tässä opinnäytetyössä pyritään vastaamaan seuraaviin kysymyksiin:

1. Minkälaisia merkityksiä sovelluksen käyttökokemuksella on?
2. Miten määrällinen tukee laadullista käyttöliittymä- ja käyttökokemustutkimuksen puitteissa?
3. Minkälaisia käyttöanalytiikan menetelmiä on hyödynnettävissä ja minkälaisia työkaluohjelmistoja on markkinoilla?
4. Miten voidaan kehittää käyttötiedon keruu-, analysointi- ja visualisointialusta, joka toimii yksityisessä verkossa ilman pilvipalvelua ja on käyttökelpoinen Qt-pohjaisten käyttöliittymäsovellusten käytön ja käytettävyyden tutkimiseen?

Kysymys 4 on päätutkimuskysymys, jonka puitteissa on, analytiikkajärjestelmän kehittämisen lisäksi, tarkoitus soveltaa ja luoda muutamia analytiikkamenetelmiä, kuten aktiivisuuskartan generointi ja käyttäjäpolkujen tunnistus, jotka hyödyntävät tekoälyä mahdollisuuksien mukaan. Analytiikan toimintaa on tarkoitus havainnollistaa oikean tuotteen käyttöliittymän tai toteutettavan demokäyttöliittymän avulla, mutta analytiikan projektointi ja käyttöönotto on rajattu pois tämän opinnäytetyön piiristä. Lisäksi laadullinen analyysi, kuten syiden etsiminen tai esimerkiksi tehtävien onnistumisprosentin kaltaisten tietojen selvittäminen jätetään analytiikkajärjestelmän käyttäjälle.

1.4 Tutkimusmenetelmät

Tässä opinnäytetyössä sovelletaan konstruktiivista tutkimusotetta, jota pohjustetaan teoreettisella pohdinnalla käyttökokemuksen merkityksestä toimeksiantajan kontekstissa ja määrällisten menetelmien roolista laadullisten menetelmien tukena sekä empiirisellä katsauksella analytiikkatyökalujen markkinatilanteeseen. Tarkoitus on soveltaa ja luoda määrällisen tutkimuksen menetelmiä, joilla luodaan tilastollinen ymmärrys faktoista numeerisen datan pohjalta esimerkiksi graafeja ja taulukoita hyödyntämällä. Laadullisen ja määrällisen tutkimuksen ominaispiirteitä on kuvattu taulukossa 1 Mcleod'ia (2023) mukaillen.

Laadullinen	**Määrällinen**
Miksi? Miten? Millainen?	Mitä? Missä? Kuinka paljon? Kuinka usein?
Käyttäytymisen ja tarkoitusten ymmärtäminen	Numeeristen faktojen tunnistaminen
Joustavampi, syvällisempi	Kattavampi, yleistettävämpi
Yksittäiset istunnot	Tilastot

Taulukko 1. Laadullisen ja määrällisen tutkimuksen ominaispiirteet (mukaillen Mcleod 2023)

Määrällinen tutkimus, joka on myös empiiristä tutkimusta, viittaa mihin tahansa tutkimukseen, joka perustuu johonkin tarkasti ja täsmällisesti mitattavaan. Laadullisella tutkimuksella puolestaan tarkoitetaan mitä tahansa tutkimusta, joka perustuu johonkin, jota on mahdotonta mitata tarkasti ja täsmällisesti. (Rutgers University 2023.)

Tämän opinnäytetyön konkreettisena lopputuloksena on sovelluskokonaisuus-artefakti, jolla voi automaattisesti mitata Qt-pohjaisen sovelluksen käytettävyyteen vaikuttavia asioita. Analytiikkajärjestelmä seuraa ja analysoi varsinaisen tuotteen käyttöä ja keskittää tulosten keräämisen palvelimelle. Kerättävä data perustuu operatiivisen käyttöliittymän ohjelmistoalustan tuottamiin tapahtumiin ja niiden pohjalta pystytään tekemään monipuolista analyysia. Järjestelmän operointiin ei tarvita erillisiä fyysisiä kytkimiä tai painikkeita. Tuotteen käyttö tapahtuu oletetusti hiiren ja näppäimistön avulla, joiden tuottamat tapahtumat saadaan kerättyä talteen. Analytiikkaa tehdään sekä päätelaite- että verkkotasolla. Analyysissa käyttödatasta lasketaan tilastollisia tunnuslukuja, säännönmukaisuuksia ja riippuvuuksia. Tuloksia on mahdollista tarkastella ensisijaisesti käyttöliittymäsovelluksen grafiikan päälle lisättynä, mutta niitä on helppo myös tulostaa tekstimuodossa konsoliin tai tiedostoon.

Analyysien tulosten esitystavat valitaan siten, että ne tuottavat mahdollisimman selkeän lopputuloksen.

2 Käyttökokemuksen merkitys

2.1 Markkinoinnissa

Brändäys ja UI/UX-suunnittelu ovat Bergin (2023) mukaan toisistaan erottamattomia asioita ja yhdessä ne määrittävät, mitä yritys tai organisaatio haluaa näyttää maailmalle luodessaan sovellusta. Berg lisää, että brändi on yrityksen ydin, joka määrittelee heidän näkemyksensä ja tarkoituksensa. Sovelluksen ulkoasun, käyttökokemuksen ja vuorovaikutuksen merkitys markkinoinnissa, lähinnä messuilla ja suurtapahtumissa, on siis ensiarvoisen tärkeää.

Kilpailu huomiosta on kovaa, eikä esimerkiksi teknologinen etumatka tai laadukkaat esitteet välttämättä takaa sen saavuttamista. Saumattoman käyttökokemuksen tarjoavan tuotteen voi ajatella helpommin herättävän osallistujien mielenkiinnon ja parantavan kuvaa sekä tuotteesta että brändistä. Esiteltävälle ohjelmistolle saattaa messuilla tulla monenlaista käyttäjää ja he saattavat käyttää ohjelmistoa täysin odottamattomalla tavalla. Tästä syystä on tärkeää, että ohjelmistossa on vain hyvin testattuja ominaisuuksia mukana, tai kokeelliset ominaisuudet ovat selvästi erotettavissa. Ohjelmiston hidas, virheellinen tai epälooginen toiminta näkyy ainakin tuotetta operoivalle käyttäjälle, mutta taustalla voi olla useampiakin uteliaita katsojia eri yrityksistä ja organisaatioista.

2.2 Loppukäyttäjän arjessa

Sovelluksen käyttökokemus vaikuttaa loppukäyttäjän arkeen monella tavalla. Käytön helppous, virheettömyys ja tuottavuus ovat tässä ensisijaisen tärkeitä. Kun käyttökokemus on sujuva ja miellyttävä, se parantaa Niemelän (2022) mukaan asiakkaan kokonaisvaltaista kokemusta, tuo hyötyä asiakkaalle ja auttaa yritystä erottautumaan kilpailijoista. Sujuva ja miellyttävä käyttöliittymä on intuitiivinen, jolla uusi tai palaava käyttäjä pystyy nopeasti tekemään perustehtäviä.

Pohjimmiltaan käyttökokemus muokkaa sitä, miten ihmiset ovat vuorovaikutuksessa teknologian, palvelujen ja jopa fyysisten ympäristöjen kanssa päivittäisessä elämässään. Se voi joko yksinkertaistaa tehtäviä, vähentää stressiä ja lisätä sitoutumista tai jos se on huonosti suunniteltu, aiheuttaa turhautumista ja haitata tuottavuutta. Alkuinnostuksen lisäksi sovelluksen pitäisi tarjota puitteet jatkuville miellyttävyyden, mielenkiinnon ja hyödyllisyyden tuntemuksille.

2.3 Tuotekehittäjien työlle

Omakohtaiseen kokemukseen perustuen on syntynyt käsitys, että tuotteen käytettävyys on tavallaan itseään ruokkiva kierre, sillä se vaikuttaa myös itse tuotteen kehittämiseen. Laajan ja monimutkaisen tuotteen yksittäinen kehittäjä on käytännössä loppukäyttäjän asemassa siinä mielessä, että hän ei tunne suurta osaa järjestelmän toimintaa syvällisesti. Jos tuotteen käytettävyys on suunniteltu hyvin, myös sen testaaminen on suoraviivaisempaa ja iteratiivisessa kehityksessä vertaispalaute on osuvampaa.

Käytettävyyttä olisi syytä pitää silmällä alusta asti, ettei jouduta negatiiviseen kierteeseen, jossa huonoja perusratkaisuja paikataan vain pienillä parannuksilla. Monimutkaisten järjestelmien käyttöliittymiä voi olla vaikea tehdä helppokäyttöiseksi ja esimerkiksi teknologiset valinnat näkyvät helposti käyttöliittymän toteutukseen asti. Tämän takia käyttöliittymään tulisi panostaa tuotekehityksen alusta asti sitä mukaa kun siihen kohdistuvat vaatimukset valmistuvat ja vakiintuvat. Uusien, käyttäjiltä tai johdolta tulleiden, vaatimusten hyväksymisessä ja toteuttamisessa pitäisi aina huomioida kokonaisuus myös käytettävyyden kannalta.

3 Käyttöliittymäanalytiikka osana tuotekehitysprosessia

3.1 Laadullisen tutkimuksen tukena

Käyttö[liittymä]analytiikan tuloksia voidaan Valkaman (2013) mukaan hyödyntää esimerkiksi toteutettaessa kognitiivisen läpikäynnin menetelmää, jota käytetään tunnistamaan käyttöliittymän mahdolliset käytettävyysongelmat simuloimalla käyttäjien ajatteluprosesseja heidän ollessaan vuorovaikutuksessa järjestelmän kanssa. Eskola (2008) kuvailee konkreettisemmin, että kognitiivinen läpikäynnin pohjaksi valitaan käyttäjän suorittama tehtävä, jota aiotaan testata. Valitusta tehtävästä huomioidaan Eskolan mukaan sen tavoite ja sen saavuttamisen edellyttämät vaiheet. Läpi käytävät tehtävät on hyvä valita esimerkiksi niiden yleisyyden perusteella, mikä varmistaa peruskäytön sujuvuuden, toteaa Eskola ja jatkaa, että tehtävätyypit on mahdollista valita myös siten, että ne käyvät läpi mahdollisimman monta testattavan tuotteen ominaisuutta.

Kognitiivisessa läpikäynnissä käyttäjän tehtävä jaetaan mahdollisimman pieniin osiin, joista jokainen on suoritettava järjestyksessä. Tämän jälkeen sovellusalan asiantuntija käy jokaisen osan läpi ja arvioi sitä neljän kysymyksen avulla: Pyrkiikö käyttäjä oikeaan tavoitteeseen eli ymmärtääkö hän kyseisen vaiheen olevan osa kokonaisuutta? Huomaako käyttäjä oikean toiminnon olevan tarjolla eli onko toiminto löydettävissä? Liittääkö käyttäjä toiminnon omaan tavoitteeseensa? Huomaako käyttäjä tehtävän etenemisen suorittaessaan oikean toiminnon eli saako hän selkeän palauteen toiminnon suorittamisesta? (Eskola 2008.)

Kehitetty käyttöanalytiikkajärjestelmä voi auttaa vastaamaan näihin kysymyksiin. Järjestelmä tunnistaa useimmin toistuvia tehtäviä eli käyttäjäpolkuja ja voi paljastaa niissä olevia tarpeettomia osatehtäviä eli askeleita. Myös kahden lähes identtisen käyttäjäpolun vertaaminen keskenään voi nostaa esille mahdollisia epäkohtia toisessa niistä. Ehkä vaikuttavinta olisi, jos usein toistuvien käyttäjäpolkujen joukosta löytyisi täysin tarpeettomia polkuja. Käyttöanalytiikan tulokset voivat myös tarjota vihjeitä esimerkiksi siitä minkä osatehtävän suorittamiseen liittyy usein poikkeavaa epävarmuutta.

3.2 Tuotekehitysprosessia ohjaavana tekijänä

Käyttöanalytiikkatyökalut voivat toimia ohjelmistokehitysprosessin moottoreina tarjoamalla käyttökelpoisia ideoita auttamaan päätöksenteon eri vaiheessa. Käyttöanalytiikkatyökalut keräävät määrällistä ja laadullista tietoa käyttäjien käyttäytymisestä, mieltymyksistä ja vuorovaikutuksesta ohjelmiston kanssa. Analysoimalla näitä tietoja kehittäjät voivat tehdä tietoisia päätöksiä ominaisuuksien priorisoinnista, suunnitteluvalinnoista ja optimointistrategioista.

Ohjelmistokehittäjien on oletettavasti helpompi omaksua käyttäjäkeskeinen lähestymistapa suunnitteluun käyttöanalytiikkatyökalujen tarjotessa näkyvyyttä käyttäjien ja ohjelmiston väliseen vuorovaikutukseen. Näin myös varmistetaan, että suunnittelupäätökset perustuvat käyttäjien todellisiin tarpeisiin ja käyttäytymiseen kehitystiimin oletusten tai mieltymysten sijaan.

Iteratiivinen kehitys helpottuu käyttöanalytiikkatyökalujen ansiosta, kun kehittäjät saavat mahdollisuuden seurata ja arvioida jatkuvasti suunnittelumuutosten vaikutusta käyttäjien käyttäytymiseen. Esimerkiksi A/B-testauksen kaltaisten ominaisuuksien avulla kehittäjät voivat testata hypoteeseja, mitata tuloksia ja jalostaa ohjelmistoa käyttäjien palautteen perusteella. Käyttöanalytiikkatyökalut tukevat myös jatkuvan parantamisen kulttuuria antamalla kehittäjille mahdollisuuden seurata keskeisiä suorituskykyindikaattoreita (KPI) ajan kuluessa ja asettaa menestyskriteerit. Näin kehittäjät voivat mitata ponnistelujensa tehokkuutta ja tunnistaa mahdollisuuksia lisäoptimointiin. Esimerkki keskeisestä suorituskykyindikaattorista voisi olla tiettyjen tunnettujen käyttäjäpolkujen suorittamiseen kulunut aika.

Tunnistamalla ohjelmiston osa-alueet, jotka ovat käyttäjille vajaakäytössä tai ongelmallisia, Käyttöanalytiikkatyökalut auttavat kehittäjiä kohdistamaan resursseja tehokkaammin. Kehittäjät voivat keskittyä ratkaisemaan kriittisimpiä ongelmia, joilla on suurin vaikutus käyttäjien tyytyväisyyteen. Käyttöanalytiikkatyökalut voivat myös tarjota tietoa ohjelmiston suorituskyvystä, mukaan lukien latausajoista, virhemääristä ja järjestelmän kaatumisista. Seuraamalla näitä mittareita kehittäjät voivat tunnistaa suorituskyvyn pullonkauloja ja priorisoida optimoinnit yleisen käyttökokemuksen parantamiseksi.

3.3 Turvakriittisen ympäristön haasteet

Tietoturvaan liittyviä kysymyksiä oletettavasti nousee esille, kun analytiikkaa sovelletaan tämän tuotteen parissa. Analytiikan käytön odotetaan kuitenkin olevan riskitöntä, koska tulokset eivät vaikuta suoraan asiakkaiden operatiiviseen toimintaan. Ylipäätään erillisverkko usein oletetaan tietoturvalliseksi ja vain työkäyttöön tarkoitetuksi ympäristöksi. Tietoturva muodostuu kysymykseksi lähinnä siinä, miten ja millaista dataa erillisverkoista viedään ulos.

Ehkä suurin haaste datan keräämisen ja määrällisen analytiikan tulosten jakelun kannalta on toiminnan rajautuminen organisaation sisäiseen verkkoon. Dataa ei voi kerätä sellaisenaan useista organisaatioista keskitettyä analyysiä varten, jolloin tulosten vertailukelpoisuus esimerkiksi kehitysympäristön ja asiakkaiden tuotantoympäristöjen välillä heikkenee. Analyysin pitää tapahtua erillisverkkojen sisällä ja tulokset ovat riippuen asiakkaasta, mahdollista tuoda tarkastettuna ja siistittyinä julkiseen verkkoon. Tällöin kyseessä on julki- ja yksityispilven muodostama hybridiratkaisu, jossa arkaluontoinen data jää yksityispilveen ja

abstrahoitu malli tuodaan julkipilveen. Yhdistetyn analyysin tuloksia ei myöskään ole mahdollista sellaisenaan siirtää asiakkaiden verkkoihin, vaan esimerkiksi käyttöä parantavat toimenpide-ehdotukset viedään sinne erikseen dokumentoituna materiaalina. Tämä luonnollisesti tekee käyttökokemuksen kehitysprosessista hitaamman ja vähemmän systemaattisen korostaen tarkan ja täsmällisen tiedonhankinnan merkitystä.

3.4 Eettisten kysymysten huomiointi

Käyttöanalytiikka voi paljastaa loppukäyttäjän työskentelytapoja, mutta sen ei tule paljastaa mitään yksityisyyden suojan alle kuuluvaa tietoa. Sovelluksen käytön edellytyksenä tulisi välttää esimerkiksi henkilötietojen käsittelyä eikä analytiikkaa pitäisi olla tarpeen integroida asiakkaan muihin valvontajärjestelmiin. Teoriassa analytiikan tuloksia voisi käyttää esimerkiksi objektiivisena työkaluna operaattorikandidaattien omaksumisnopeuden mittaamisessa, mutta sen perusteella olisi todennäköisesti vaikea päätellä mitään esimerkiksi työntekijän soveltuvuudesta työhön.

Suomen laissa on säädetty muun muassa henkilötietojen käsittelyyn, henkilö- ja soveltuvuusarviointitesteihin ja yhdistettyyn valvontaan liittyviä pykäliä, jotka takaavat yksityisyyden suojaa turvaavia perusoikeuksia työelämässä (Finlex 2004). Näitä tulee pitää mielessä myös analytiikkajärjestelmän kehitystyössä ja varmistaa käyttäjän yksityisyydensuojan säilyminen. Parhaiten käyttäjän yksityisyydensuoja säilyy, kun mitään käyttäjää yksilöiviä tietoja ei tallenneta.

3.5 Projektointi ja jalkauttaminen

Käyttöliittymä- ja kokemusanalyysin tulosten projektointi ja jalkauttaminen olisi tärkeä toteuttaa, jotta ne eivät jäisi pelkäksi sanahelinäksi. Se on kuitenkin helpommin sanottu kuin tehty, koska tämän kaltaiset muutokset vaikuttavat yleensä tuotteen kehittäjä- ja käyttäjäkuntien ko’oista riippuen lukuisten ihmisten työntekoon. Määrällisen analytiikan työkaluilla tuotettu täsmällinen tieto on kuitenkin vahva vaikutin muutoskohteiden ymmärtämisessä ja muutostarpeen hyväksymisessä.

Projektointi ja jalkauttaminen tulisi aloittaa tunnistamalla tärkeimmät käyttöliittymä- ja kokemusanalyysistä saadut löydökset. Näitä voivat olla esimerkiksi käytettävyysongelmat tai käyttäytymismallit. Tämän jälkeen tunnistetut löydökset tulisi arvioida laadullisesti ja priorisoida niiden käyttökokemukseen ja kehityssuunnitelmaan kohdistuvien vaikutuksen perusteella. Huomio tulisi suunnata sellaisten ongelmien ratkaisemiseen, joilla on suurin vaikutus käyttäjien tyytyväisyyteen, säilyttämiseen ja projektin kannattavuuteen. Priorisoidut näkemykset tulisi jakaa toteutettavissa oleviksi tehtäviksi tai suosituksiksi kehitystiimille.

Jokainen tehtävä olisi määriteltävä selkeästi, ja siinä olisi oltava erityiset, mahdollisesti mittauksiin perustuvat, tavoitteet ja tulokset. Tämän jälkeen tulisi määrittää vastuut tunnistettujen tehtävien toteuttamisesta asianmukaisille tiimin jäsenille, mukaan lukien suunnittelijoille, kehittäjille ja tuotepäällikölle. Jokaisen tehtävän osalta olisi varmistettava, että sillä on selkeä omistajuus ja vastuu.

Suositellut muutokset tai parannukset ohjelmistoon tulisi toteuttaa käyttöliittymä- ja kokemusanalyysistä saatujen löydösten perusteella. Tämä voi sisältää suunnittelun iteraatioita, ominaisuuksien parannuksia, suorituskyvyn optimointeja tai käytettävyyden parannuksia. Kun muutokset on toteutettu, tulisi suorittaa esimerkiksi käytettävyystestaus tai A/B-testaus niiden tehokkuuden vahvistamiseksi. Todellisilta loppukäyttäjiltä tulisi kerätä palautetta, joka auttaa arvioimaan ovatko muutokset ratkaisseet tunnistetut ongelmat ja parantaneet yleistä käyttökokemusta.

Testauksesta kerättyä palautetta tulisi käyttää viemään valmisteltuja muutoksia eteenpäin. Käyttäjien käyttäytymistä ja palautetta tulisi seurata jatkuvasti uusien parannusmahdollisuuksien löytämiseksi. Käyttöliittymä- ja kokemusanalyysin opit ja tulokset sekä niistä johtuvat ohjelmistoon tehdyt muutokset tulisi dokumentoida. Tämä dokumentaatio toimisi arvokkaana viitemateriaalina tulevia iteraatioita varten antaen tietoa myöhempien kehitysjaksojen päätöksentekoa varten. Jatkuvan parantamisen kulttuuria tulisi edistää kehitystiimin sisällä kannustamalla sitä toteuttamaan jatkuvaa käyttökokemuksen analysointia ja optimointia. Käyttäjäpalautteen ja tietopohjaisen päätöksenteon merkitystä tulisi korostaa ohjelmistokehitysprosessin muutostyössä.

4 Käyttöliittymätutkimuksen määrällisistä tutkimusmenetelmistä

4.1 Määrälliset menetelmät laadullisten menetelmien tukena

Data-analytiikka on keskeisessä roolissa tämän opinnäytetyön toteutuksessa, jossa pyritään löytämään ja soveltamaan hyödyllisiä analyysimenetelmiä laadullisen käyttöliittymä- ja kokemustutkimuksen tueksi. Data-analytiikassa datamassaa pyritään Stedmanin (2022) mukaan muokkaamaan, yhdistämään, analysoimaan ja raportoimaan toimeksiantajaa hyödyttävällä tavalla. Prosessin tarkoituksena voi olla poikkeavuuksien löytämisessä käyttäytymisestä, tunnistaa suuntauksia, tai ennustaa tulevaa käyttäytymistä, Stedman lisää. Data-analyysin vaiheet ovat Taanilan (2023) mukaan datan hankkimisen ja esikäsittelyn jälkeen tyypillisesti seuraavat: kuvaileva analytiikka, selittävä analytiikka ja ennakoiva analytiikka.

Kuvaileva analytiikka voi muodostua esimerkiksi lukumäärä- ja prosenttiyhteenvetojen laskemista kategorisille muuttujille, luokiteltujen jakaumien ja tilastollisten tunnuslukujen laskemista määrällisille muuttujille, prosenttimuutosten laskemista aikasarjoille, aikasarjojen tarkastelemista viivakaavioina ja liukuvien keskiarvojen esittämistä aikasarjojen yhteydessä. Kuvailun tuloksia suositellaan visualisoitavaksi ja havainnollistettavaksi laadukkailla taulukoilla ja kaavioilla.

Selittävä analytiikka voi muodostua esimerkiksi tilastollisten tunnuslukujen vertailusta eri ryhmissä, kategoristen muuttujien riippuvuuden tarkastelusta ristiintaulukoimalla, määrällisten muuttujien välisten korrelaatioiden tarkastelusta sekä havaittujen erojen ja riippuvuuksien tilastollisen merkitsevyyden tarkastelusta.

Ennakoiva analytiikka tarkoittaa data-analytiikan piirissä lähinnä koneoppimista. Koneoppimisen malleilla voidaan luokitella ja ennakoida määrällisen muuttujan arvoja. Koneoppiminen perustuu siihen, että kone oppii käytettävän mallin parametrit olemassa olevasta datasta ja tämän jälkeen mallia voidaan soveltaa uuteen dataan. Koneoppimisalgoritmit voidaan luokitella ohjattuun oppimiseen, ohjaamattomaan oppimiseen ja vahvistusoppimiseen.

Ohjatussa oppimisessa algoritmi opetetaan opetusdatalla ja näin muodostunutta mallia käytetään tunnistamaan tiedettyjä ominaisuuksia uudesta datasta. Ohjaamattomassa oppimisessa ei ole valmiina oikeita luokkia, vaan tehtävänä on löytää datasta jonkinlainen rakenne, esimerkiksi samankaltaisten tapausten ryhmiä tai ryppäitä (engl. clusters), tai esitysmuoto, jossa data voidaan esittää muutaman tärkeimmän muuttujan tai ulottuvuuden avulla. Vahvistusoppimisessa algoritmi toimii yleensä monimutkaisessa ympäristössä ja päivittää mallia saamansa palautteen mukaan. Käytännössä näiden oppimiskategorioiden

rajat eivät ole kovin selkeitä, ja yksittäistä koneoppimisongelmaa tai -menetelmää saattaa olla vaikea lokeroida. (Elements of AI 2024.)

4.2 Käyttötapaukset ja käyttäjäpolut

Browne (2024) määrittää käyttötapauksen koostuvan yhdestä tai useammasta käyttäjäpolusta ja toteaa myös käyttäjäpolun kuvaavan, miten käyttäjä etenee digitaalisen asiakaspolun läpi aloituspisteestä tehtävän suorittamiseen ja palvelusta poistumiseen. Tätä määritelmää Baskanderi (2017) tarkentaa toteamalla, että käyttäjän polku ei yleensä etene lineaarisesti, vaan se voi haarautua useita kertoja. Mahdollisten vaihtoehtojen määrittely auttaa Baskanderin mukaan havaitsemaan palvelun mahdolliset epäjohdonmukaisuudet ja parantamaan käyttäjäkokemusta.

Käyttäjäpolkujen ymmärtämisen ja optimoinnin voi ajatella olevan ratkaisevan tärkeitä saumattoman ja intuitiivisen käyttöliittymän suunnittelussa (Adobe 2022b). Esimerkiksi käyttötapauksissa tarvittava osoittimen siirtymismatka saattaa antaa vihjeen siitä onko näkyvissä oleva tieto järjestetty loogisesti ja käyttökelpoisesti. Käyttäjäpolkujen mitatuista ominaisuuksista ja arvoista voi määrällisen tutkimuksen puitteissa myös seurata muutosta, joka tapahtuu siirryttäessä ohjelmaversiosta tai ympäristöstä toiseen.

Avainkomponentit käyttäjäpoluissa ovat tunnetusti käyttäjän vuorovaikutuksen aloituskohta, vuorovaikutus ja toiminta, valintakohdat, palaute ja hyväksyntä sekä päätöskohta. Aloituskohta voi olla painike, näkymä tai sovelluksen käynnistäminen. Vuorovaikutus ja toiminta voi olla sarja vaiheita, joita käyttäjät suorittavat saavuttaakseen tavoitteensa. Näitä ovat esimerkiksi painikkeiden napsauttaminen, lomakkeiden täyttäminen, valintojen tekeminen, sivujen välillä liikkuminen ja muut sovelluksen edellyttämät toimet. Valintakohdat ovat hetkiä, jolloin käyttäjien on tehtävä valintoja tai päätöksiä, jotka vaikuttavat käyttäjäpolun suuntaan. Päätöspisteet voivat sisältää vaihtoehtojen valitsemisen, syötteen antamisen tai valinnan eri polkujen välillä.

Tässä opinnäytetyössä näitä käyttäjäpolun komponentteja kutsutaan askeliksi, eikä käyttäjäpolku ei välttämättä pääty palvelusta poistumiseen, kuten Browne (2024) määrittelee. Palaute, jonka käyttäjät saavat toimintojen aikana ja sen jälkeen, voi sisältää viestin onnistumisesta tai muuta tietoa, joka vahvistaa tai ohjaa käyttäjää vuorovaikutuksen perusteella.

Ohjelmistosuunnittelijat ja UX-ammattilaiset käyttävät yleensä vuokaavioita tai visuaalisia esityksiä vuorovaikutusjärjestyksen kartoittamiseen ja analysoimiseen. Nämä kaaviot voivat auttaa tunnistamaan mahdollisia kipukohtia ja väärinymmärryksiä tai muuten löytämään mahdollisuuksia käyttökokemuksen parantamiseksi. Esimerkiksi pitkien ja usein toistuvien käyttäjäpolkujen laadulliseen tutkimiseen saattaa kannattaa panostaa siltä varalta, että

sama työ ymmärrettäisiinkin tehdä nopeammin tai vähemmillä määrillä askelia. Käyttäjäpolun optimointiin voi kuulua suorittamispolun yksinkertaistaminen ja virtaviivaistaminen ja sen varmistaminen, että käyttöliittymä ohjaa käyttäjää kohti todennäköisintä tavoitetta.

4.3 Käyttäjäpolkujen tunnistaminen automaattisesti

Käyttäjäpolkujen tunnistaminen automaattisesti on oletettavasti tärkeää, koska tietomäärä on potentiaalisesti suuri ja tulokset ovat sitä hyödyllisempiä, mitä kattavammin koko käyttäjäkunta ja käyttöliittymä on huomioitu analyysissä. Käyttäjäpolkujen, kuten käyttöliittymän rakenteenkin, selvittäminen pyritään tekemään automaattisesti seuraamalla käyttöliittymäympäristön tapahtumia. Malli käyttäjäpolusta muodostetaan tapahtumista, jotka ilmoittavat tapahtuma-ajan ja tapahtumatyypin lisäksi näppäimistösyötteen ja hiirivalinnan kohteena olevan käyttöliittymäkomponentin nimen. Käyttäjäsyötteiden tapahtumavirrasta pyritään tunnistamaan automaattisesti eri tasoilla mahdollisesti sisäkkäinkin toistuvia kuvioita.

Tunnistettuja käyttäjäpolkuja voidaan tarkastella tarvittaessa yksittäin laadullisesti, mutta niiden alku- ja loppukohtien automaattiseen tunnistamiseen tarvitaan myös laadullista tarkastelua. Tunnistetut käyttäjäpolut voivat kattaa useita todellisia käyttäjäpolkuja tai ne voivat olla osa laajempaa todellista käyttäjäpolkua - näiden toteamiseksi on kuitenkin tehtävä lisäksi laadullista tutkimustyötä. Pitkiä ajallisia taukoja ei oletettavasti voi käyttää käyttäjäpolkujen erottamiseen, koska niitä voi muodostua käyttäjäpolun sisälle esimerkiksi käyttäjän odotellessa muutosta, joka tapahtuu operatiivisessa datassa.

Tarkoitus on selvittää käyttäjäpolkujen rakenne, johon sisältyy esimerkiksi hiiren osoittimen kohdistaminen käyttöliittymäkomponentteihin. Sen sijaan käyttäjän tietosisältöä, esimerkiksi kirjoitettua tekstiä, ei näissä analyysimenetelmissä haluta käsitellä, vaan se pyritään jättämään pois mahdollisimman varhaisessa vaiheessa. Käyttäjäpolku voi sisältää operatiivisesta datasta suoraan riippuvia askeleita, esimerkiksi käsin tehtyjä karttaobjektien valintaa. Tällainen tieto ei todennäköisesti kuitenkaan erottuisi käyttäjäpolun tilastollisissa tuloksissa, kunhan dataa on kerätty riittävästi. Tällaisten tietojen erottumisen estää käytännössä se, että esimerkiksi skene- ja karttaobjektit on tavallisesti toteutettu eri tekniikalla kuin normaalit käyttöliittymäkomponentit, jolloin interaktio niiden kanssa ei tuota automaattisesti tapahtumia analyysiin.

4.4 Käyttöliittymän suorituskyvyn mittaamisesta

Käyttöliittymän suorituskyvyllä tarkoitetaan yleensä sitä, kuinka hyvin käyttöliittymä täyttää tietyt kriteerit esimerkiksi reagointikykyyn, nopeuteen, tehokkuuteen ja yleiseen käyttökokemukseen liittyen. Varsinkin näiden kahden jälkimmäisen mittaaminen tarkasti ja

absoluuttisesti lienee mahdotonta, mutta tässä lopputyössä pyritään kuitenkin muodostamaan johtopäätöksiä ohjaavia tunnuslukuja, joiden kehittymistä on mahdollista tarkastella sovellusversioiden välillä. Yksi mitattava asia on reagointiviive, jolla käyttöliittymä vastaa käyttäjän syötteisiin ja vuorovaikutukseen. Matalan viipeen voi olettaa indikoivan sujuvaa käyttökokemusta ja tehokasta käyttöä, mutta toisaalta pitkä odotusaika ei välttämättä tarkoita ongelmaa - käyttäjäpolussa voi olla tauko esimerkiksi silloin, kun käyttäjä tarkkailee jotain reaaliaikaista datanäyttöä seuraavaa toimintaansa varten.

Hyväksyttävistä maksimiviiveistä käyttäjän toimenpiteiden ja järjestelmän vastausten välillä on Nielsenin (1993) mukaan vuosikymmeniä samana pysynyt neuvo:

> **0,1 sekuntia** on suurin piirtein raja sille, että käyttäjä tuntee järjestelmän reagoivan välittömästi, eli mitään erityistä palautetta ei tarvita, paitsi tuloksen näyttäminen.
>
> **1,0 sekuntia** on suunnilleen raja, jolla käyttäjän ajatuskulku pysyy keskeytyksettä, vaikka käyttäjä huomaa viiveen. Normaalisti erityistä palautetta ei tarvita yli 0,1 mutta alle 1,0 sekunnin viiveillä, mutta käyttäjä menettää tunteen toimia suoraan datalla.
>
> **10 sekuntia** on suurin piirtein raja, johon asti käyttäjän huomio voidaan pitää. Pidemmillä viiveillä käyttäjät haluavat suorittaa muita tehtäviä odottaessaan tietokoneen suorittaman tehtävän valmistumista, jolloin heille tulisi antaa tietoa edistymisestä. Palaute viiveen aikana on erityisen tärkeää, jos vasteajan odotetaan vaihtelevan suuresti, koska käyttäjät eivät muutoin tiedä lainkaan mitä odottaa.

Koponen (2019) tarkentaa tätä viittaamalla tutkimusnäyttöön, joka osoittaa jo noin puolen sekunnin viiveen käyttäjän toimen ja siitä seuraavien tapahtumien välillä vaikuttavan huomattavan haitallisesti vuorovaikutteisen visualisoinnin käytettävyyteen. (Koponen ym. 2019.)

Odotusajat sovelluksissa voivat vaikuttaa merkittävästi käyttökokemukseen ja siten yleiseen tyytyväisyyteen sovellusta kohtaan. Liian pitkät odotusajat voivat aiheuttaa käyttäjien turhautumista ja tyytymättömyyttä. Käyttäjät tunnetusti arvostavat reagoivaa ja nopeaa vuorovaikutusta. Vaikka sovellus suorittaa teknisesti tehtäviä taustalla, käyttäjät voivat kokea sovelluksen hitaana ja tehottomana, jos käyttäjät kokevat merkittäviä viiveitä käyttäjäpolun etenemisessä. Käyttäjillä on usein tiettyjä tehtäviä tai tavoitteita sovellusta käyttäessään, joiden suorittamisen tehokkuutta pitkät odotusajat voivat haitata. Järjestelmän

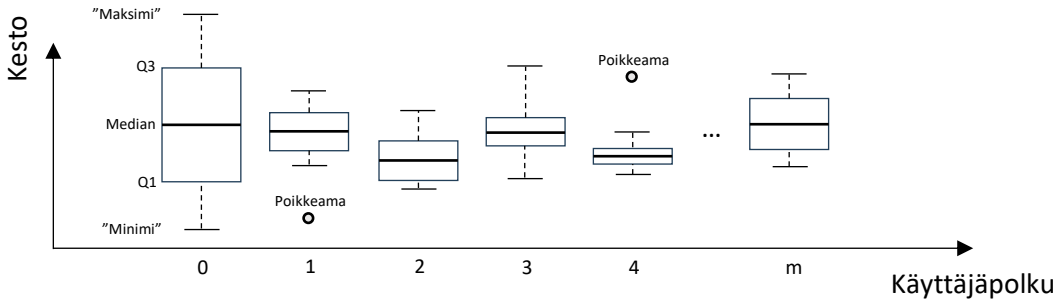
reagointiviiveitä käsitellään tässä opinnäytetyössä tarkemmin samalla kun analysoidaan käyttäjäpolkujen sisäisiä viiveitä.

4.5 Käyttäjäpolun kesto

Käyttäjäpolun kesto on tiedossa, kun käyttäjäpolun kaikki askeleet on tunnistettu. Tarkan aloitus- ja lopetuskohdan määrittäminen ei ole tarpeen käyttäjäpolun keston määrittämiseksi. Erityisen lyhyt tai pitkä kokonaisaika saattaa merkitä epäonnistunutta tehtävää tai keskeytystä ulkoisen syyn takia, varsinkin, jos kyseinen käyttäjäpolku ei ole toistunut useasti. Pitkä kokonaisaika voi viitata tarpeettoman monimutkaiseen tai epäkäytännölliseen käyttäjäpolkuun (Nawrocki 2022) – varsinkin toteutuessaan useammin kuin toisinaan.

Samansisältöisten käyttäjäpolkujen keston suuren hajonnan voi olettaa kertovan polkuun liittyvien, mutta käyttäjästä riippumattomien, odotusaikojen suuresta vaihtelusta tai siitä, että käyttöliittymä ei tue riittävästi käyttötapauksen toteuttamisen sujuvaa etenemistä. Tämä riittämätön tuki on kiinnostava käyttökokemuksen kehittämisen kannalta, joten se olisi hyvä olla erotettavissa käyttäjästä riippumattomista odotusaikojen suuresta vaihtelusta. Tällaisia tapauksia on mahdollisesti syytä tutkia lisää laadullisen tutkimuksen menetelmin.

Käyttäjäpolkujen kestoja saattaisi olla hyödyllistä seurata uuden käyttöliittymän käyttöönottamisen jälkeen, jolloin mahdollisesti käyttäjäpolkujen intuitiivisuus ja sisäistyminen käyttäjäkunnalle tulisi mitatuksi. Tämä luonnollisesti edellyttäisi käyttäjäpolkujen tunnistamisen deterministisyyttä käyttöliittymän peräkkäisten versioiden välillä. Tunnistettujen käyttäjäpolkujen kestojen tilastollisia tunnuslukuja voidaan kuvata esimerkiksi seuraavasti:



Kuvio 1. Käyttäjäpolkujen kokonaisajat

4.6 Hiirellä ja näppäimistöllä tehdyn työn määrä

Hiiri- ja näppäimistötyön määrä eri käyttäjäpoluissa on mitattavissa. Pitkä hiiren osoittimen kulkeman matkan tai matka-ajan voi olettaa kertovan esimerkiksi liian hajallaan olevista kontrolleista. Suuri hajonta hiiren osoittimen kulkemassa matkassa tai matka-ajassa

saattaa puolestaan kertoa käytön epävarmuudesta. Yksittäisten näppäimistöpainallusten (todennäköisesti pikanäppäinpainalluksia) määrä voi osaltaan mitata käyttäjäpolun monimutkaisuutta. Samanlaisena toistuva näppäinpainallusten sarja voidaan mahdollisesti laskea osaksi käyttäjäpolun rakenteeseen, mutta sen voisi epäsäännöllisesti tapahtuvana tulkita kertovan esimerkiksi käyttäjän turhautumisesta tai epäonnistuneista yrityksistä.

Näppäinpainallusten sarja saattaa myös olla operatiivista tietoa, jota syötetään tekstikenttään. Käyttäjäpolkuja tunnistettaessa tällainen sarja onkin tarpeellista tunnistaa ja tietosuojan säilyttämiseksi se esitetään pelkistettynä vain aloitusmerkin sisältävänä syötetapahtumana, josta nähdään kirjoittamisen aloitusaika ja/tai mahdollinen pikanäppäimen käyttö. Tämän opinnäytetyön analytiikka keskittyy kuitenkin enemmän hiirellä kuin näppäimistöllä tehdyn työn mittaamisen ja analysointiin.

4.7 Tehtävien onnistuminen

Tehtävän onnistumisen mittaamiselle automaattisesti käyttödatasta ei odoteta löytyvän keinoa, koska data ei sisällä todellisia tietoja käyttäjän tavoitteesta. Sen sijaan toteutuneita käyttäjäpolkuja voidaan jälkikäteen verrata suunniteltuihin käyttäjäpolkuihin laadullisen tutkimuksen puitteissa. Käyttäjäpolkujen onnistumisastetta voi tällöin arvioida esimerkiksi binaarisesti - kuinka suuri osuus yrityksistä onnistui sataprosenttisesti tai vaihtoehtoisesti kuinka pitkälle suunniteltua käyttäjäpolkua päästiin keskimäärin.

Epäonnistumisen tunnistamisen voisi ajatella olevan helpompaa kuin onnistumisen tunnistamisen, koska epäonnistumiset ovat oletettavasti paljon harvinaisempia kuin onnistumiset. Käyttöliittymän tuottama virheilmoituksen antaminen voisi käytännössä indikoida tehtävän epäonnistumista, mutta käyttöliittymän ohjelmistoalusta ei tavallisesti tuota siitä tapahtumaa. Potentiaalisia epäonnistumisia voidaan mahdollisesti löytää tutkimalla tunnistettujen käyttäjäpolkujen luonnetta. Poikkeavia käyttäjäpolkuja voi olla hyödyllistä vertailla automaattisesti säännöllisiin käyttäjäpolkuihin ja tunnistaa tilanne, jossa poikkeava käyttäjäpolku on alkanut oikein, mutta keskeytynyt jostain syystä. Sisällöltään tai kestoltaan erittäin poikkeavia käyttäjäpolkuja saattaa kannattaa syiden selvittämiseksi tutkia lisää laadullisen tutkimuksen menetelmin.

Harvinainen käyttäjäpolku, jossa toistuu erityisen tiheään saman näppäimen painallus tai hiiren ravistelu, voi kertoa epäonnistumisesta ja/tai käyttäjän turhautumisesta (”rage-click”), jota myös voisi olla hyvä tutkia tarkemmin laadullisen tutkimuksen puolella. Esimerkkejä tällaiseen tilanteeseen johtaneista syistä voisi olla tahmainen käyttöliittymä, painikkeet, jotka eivät toimi odotetulla tavalla, grafiikat, jotka näyttävät painikkeilta, mutta ovatkin vain tekstiä.

4.8 Lämpökartat

Lämpökartta on graafinen esitys tiedoista, joka näyttää tietyn määritteen tiheyden [tai intensiteetin] värin avulla. UX-suunnittelun yhteydessä lämpökarttoja käytetään osoittamaan, missä kohdin käyttäjät ovat vuorovaikutuksessa verkkosivuston tai sovelluksen kanssa. Niitä voidaan käyttää klikkausten, hiiren liikkeiden, vieritysten ja muun tyyppisten käyttäjien vuorovaikutusten seuraamiseen. (DesignGuru 2023.)

Jos kuva on tuhannen sanan arvoinen, lämpökartta on tuhannen oivalluksen arvoinen. Tuotteen ulkoasu voi näyttää upealta sinänsä, mutta lämpökarttojen avulla on mahdollista osoittaa kuinka hyvin ne todella toimivat. Joskus käyttäjät klikkaavat elementtejä (esimerkiksi kuvia tai otsikoita), joiden he odottavat olevan toiminnallisia. Nämä voivat olla virheellisiä klikkauksia, mutta lämpökartat keräävät tietoja riittävästä määrästä käyttäjistä, jotta poikkeamat voidaan jättää huomiotta ja esittää geneerisiä klikkausmalleja käyttäjäkunnan osalta. Tarkastelemalla klikkausten lämpökarttaa on mahdollista havaita virheelliset napsautukset ja korjata ongelman lisäämällä siihen toimintoja tai muokkaamalla merkityksettömiä elementtejä, jotta niistä tulee vähemmän klikattavia. (Hotjar 2023.)

Hiiren klikkausten ja osoittimen liikkeen lämpökartat toimivat pikemminkin tekojen takana olevan psyyken havainnoinnin kuin pelkän valintakäyttäytymisen havainnoinnin apuna. Vaikka muut lämpökarttatyypit seuraavat ja edustavat vierailijoiden tekemiä valintoja/toimintoja paikan päällä, hiiren lämpökarttojen laajuus ulottuu tätä pidemmälle, jotta voidaan ottaa huomioon valintojen/toimintojen taustalla oleva prosessi. Tämä tarkoittaa sitä, että jokaisessa tehdyssä valinnassa hiiren liike kohti tai poispäin muita käytettävissä olevia vaihtoehtoisia toimintoja osoittaa, että myös näitä vaihtoehtoja on harkittu päätöksentekoprosessin aikana. Hiiren liikkeet auttavat jäljittämään koko prosessin, joka mahdollistaa suurempivaikutteisten muutosten tekemisen käyttöliittymään tai tuotteeseen. (Shanaz 2023.)

Lämpökartan käyttötarkoitus on siis käyttöanalytiikassa lähinnä mahdollistaa kuumien pisteiden tunnistaminen ja niiden merkityksen selvittäminen. Oleellista on myös pohtia mitä vertailuja halutaan mahdollistaa (Koponen ym. 2019, 25), koska kuvion sanoma syntyy vertailusta (Koponen ym. 2019, 26, Vesa Kuusela). Kartan värivoimakkuustason dynaamisuus sekä vertailtavuus istuntojen, ohjelmistoversioiden ja järjestelmien välillä parantavat oletettavasti lämpökartan käyttökelpoisuutta.

5 Käyttöliittymätutkimuksen tunnetuimmat työkaluohjelmistot

5.1 Yleisimmät toiminnot

Käyttöanalytiikan työkaluja löytyy yleisimmin web- ja mobiiliympäristöihin ja työkalujen tarjoama ominaisuusvalikoima vaihtelee suuresti. Taulukko 2 esittää joitakin suosituimpia käyttöanalytiikan työkaluja ja niiden ominaisuuksia tarkasteltuna vuosien 2023 ja 2024 vaihteessa.

	Sovelluksen tyyppi	**Istunnon toistaminen**	**Lämpökartta**	**Kehittyneempi AI-pohjainen analytiikka**
Adobe Analytics	Web, mobile	×	×	✓
Amplitude	Web, mobile	×	×	✓
Contentsquare	Mobile	×	✓	✓
Google Analytics	Web, mobile	×	✓	✓
Hotjar	Web	✓	✓	×
Mixpanel	Web, mobile	×	×	✓
Qt Insight	Desktop, mobile	×	×	×
Smartlook	Web, mobile	✓	✓	×
UXCam	Mobile	✓	✓	×

Taulukko 2. Yleisimpien työkaluohjelmistojen perustoiminnot

Jonkinlaiset käyttäjäpolku- tai suppiloraportit löytyvät yleensä kaikista työkaluista. Istunnon toistaminen vaatii tapahtumien yksityiskohtaista tallentamista, jonka takia sen toteuttaminen ei ole kaikissa konsepteissa mahdollista. Monille ohjelmistoalustoille ei näytä olevan tällaisia analytiikkatyökaluja - näistä esimerkkeinä Gtkmm, wxWdigets, Ultimate++, DearImGUI, .NET/WPF, UWP, Java Swing ja Java FX.

5.2 Tekniset ominaisuudet ja rajoitteet

Analytiikkatyökaluista QT Insight on C++ kielellä toteutettujen käyttöliittymäsovellusten analytiikkatyökalujen joukossa harvinaisuus ja se onkin käytännössä ollut ainoa mahdollinen valinta käytettäväksi Qt-pohjaisten käyttöliittymien käytön tutkimiseen.

Qt Insight

Qt Company kuvaa sovellusta seuraavasti:

”Qt Insight on sovellusanalytiikkaratkaisu, joka on suunniteltu tarjoamaan todellista tietoa siitä, miten ihmiset käyttävät sovellusta tai laitetta. Se luotiin antamaan näkemyksen sovelluksen tai laitteen suorituskyvystä, käytöstä ja käyttäjätiedoista, joka ei ehkä muuten olisi saavutettavissa." (Qt Group 2023b)

Qt Insight -tuotteen nykyisiä ominaisuuksia ovat muun muassa käytetyimpien interaktioiden ja ominaisuuksien tunnistaminen, käytön vertailu eri käyttäjäryhmien/sovellusversioiden välillä (A/B testing), käyttäjäpolkujen tunnistaminen, virhetilanteisiin päättyneiden käyttäjäpolkujen tunnistaminen, tilastotiedon tarjoaminen ohjelman ja näkymien käytöstä (esimerkiksi käyttöaste, käyttäjämäärä ja käytön säännöllisyys), tarkka henkilökohtainen käytön jäljittäminen ja tulosten monipuolinen suodattaminen.

Qt Insight -tuotteen etuja ovat ainakin analytiikan skaalautuvuus globaaliin mittakaavaan asti, uusien ominaisuuksien ”ilmaisuus”, moderni Web-käyttöliittymä ja analytiikkatoiminnallisuuden laajennettavuus. Tuotteen rajoitteita puolestaan ovat oman analytiikan kehittämisen mahdottomuus ilman Qt:n sisäistä rajapintaa, paikallisen pilvipalvelun puuttuminen ja pilvipalvelun AWS-riippuvuus, analytiikkatoiminnallisuuden keskeneräisyys ja suppeus, AI-toiminnallisuuden oleminen vasta suunnitteilla (Haantie 2023), tuloksiin sidotun käyttöliittymägrafiikan (esimerkiksi lämpökartan) puuttuminen ja toistaiseksi kyseenalainen käyttäjän tietosuojan toteutumisen taso.

6 Työssä käytetyt ja kehitetyt analyysimenetelmät

6.1 Yleistä

Yksi kehitettävän analytiikkajärjestelmän keskeisistä suunnitteluperiaatteista on ollut, että asiakastyöasemalta lähetetään ulos vain käytön tilastotietoa käyttöliittymän tapahtumadatan sijaan. Tällöin etuna on, että käyttäjän yksittäisten valintojen ajanhetket tai käyttäjän/työaseman yksilöivää tietoa ei poistu loppukäyttäjän työasemalta. Varjopuolena on esimerkiksi se, että eri istuntojen tilastollisia kvartiilitietoja ei ole mahdollista yhdistää, joten niille lasketaan vain estimaatit visualisoinnissa. Kvartiilitietojen puuttumista kompensoidaan varhaisella poikkeamatunnistuksella (engl. outlier detection), joka parantaa minimi-, maksimi- ja keskiarvojen normaalijakaumanmukaisuutta.

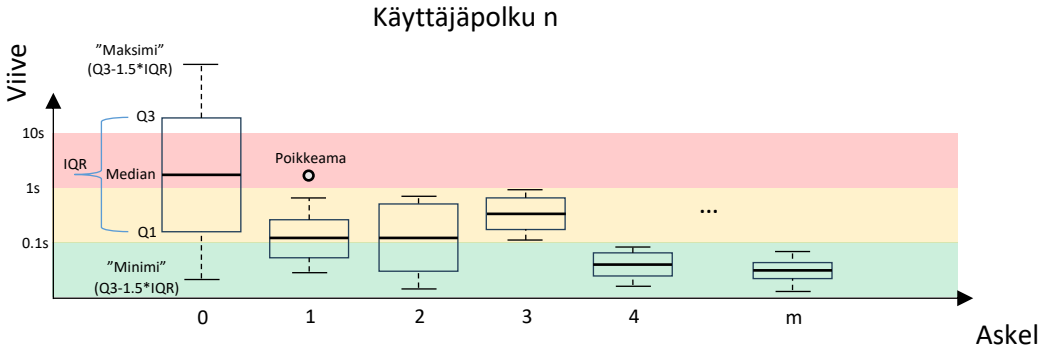
Analytiikka perustuu pitkälti käyttäjän toimenpiteisiin ja valintoihin kohdistuviin mittauksiin, jonka takia näiden mittaamisen tarkkuudella on merkitystä. Esimerkiksi hiiren osoittimen sijainnin päivitystaajuus vaikuttaa sekä lämpökarttojen tarkkuuteen että osoittimen kulkumatkojen mittauksiin. Tämä päivitystaajuus riippuu käyttöjärjestelmän hiiriajurille annetusta asetuksesta, joka on tyypillisesti välillä 125-1000Hz. Analyysin tulosten parantamiseksi asetuksen arvoa voi pyrkiä nostamaan.

Tässä kappaleessa kuvatut menetelmät ovat osa opinnäytetyön toteutusta ja kehitetty sovelluksen analytiikkalisäosan toimintaa silmällä pitäen. Yleisiä tilastotieteen peruskaavoja on luonnollisesti käytetty osina laajempia kokonaisuuksia, joissa toteutetaan kuvailevaa ja selittävää analytiikkaa. Ennakoivan analytiikan käyttäminen jätetään jatkotutkimusaiheeksi esimerkiksi käyttäjän tekoälyä hyödyntävien avustamismahdollisuuksien tutkimisen muodossa. Vaikka menetelmät ovat kehitetty Qt-työpöytäsovelluksen tutkimista varten, ne soveltuvat todennäköisesti konseptitasolla minkä tahansa muunkin tapahtumapohjaisen käyttöliittymäkehyksen päälle tehdyn sovelluksen tutkimiseen.

6.2 Käyttäjäpolun askelten kestot

Käyttäjäpolun askelten kestot ovat mitattavissa käyttäjäpolun muodostavien tapahtumien välisistä aikaeroista. Tuotteesta johtuvia viiveaikojen voi olettaa toistuvan systemaattisesti ja siksi osuvan pääasiassa samojen käyttäjäpolun askelten kohdalle. Viiveiden vaihtelun voi olettaa johtuvan myös esimerkiksi käsiteltävän datan määrästä; käyttöliittymä voi hetkittäin olla reagoimaton, kun sovellus käsittelee suurta tietomäärää tai tekee monimutkaisia laskutoimituksia, mutta tämä on luettavissa osaksi käyttöpolkua, eikä siksi vaadi eliminointia.

Lisäksi vaihtelevia viiveitä voi koitua lukemattomista syistä ja niiden lähteitä voi olla vaikea erottaa toisistaan ilman laadullista tarkastelua, jonka takia mitatuista luvuista ei välttämättä kannata tehdä laajempia johtopäätöksiä. Laadullisen tutkimuksen tueksi niistä voisi kuitenkin olla hyödyllistä tuottaa kuvaajia ja tunnuslukuja ja esimerkiksi suuri hajonta käyttäjäpolun sisäisissä viiveissä voi mahdollisesti kertoa käyttäjäpolun seuraavan askeleen löytämisen vaikeudesta. Tunnistettujen käyttäjäpolkujen sisäisten viiveiden tilastollisia ominaisuuksia voidaan visualisoida kuvion 2 mukaisesti.



Kuvio 2. Käyttäjäpolun viiveet

6.3 Hiiren osoittimen kulkema skaalattu matka

Käyttöliittymäkomponenttien koko voi analytiikan näkökulmasta ohjelman suorituksen aikana muuttua käytännössä ennakoimattomasti ja hiiren osoitinta viedään useiden komponenttien yli yhdessä käyttäjäpolun askeleessa.

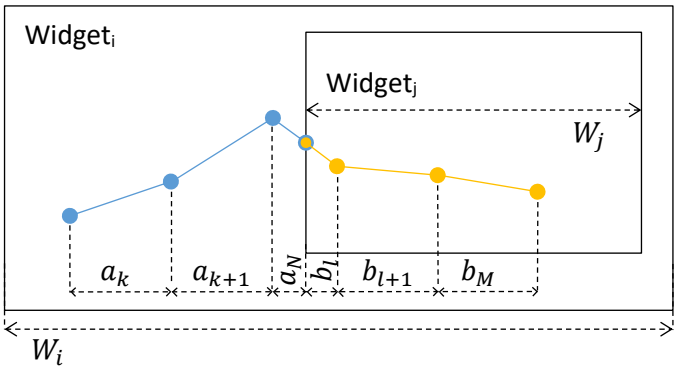
Qt:ssa käyttöliittymäkomponenttien kokoa määrittää asettelunhallinta (engl. layout management), joka huolehtii lapsikomponenttien sijoittamisesta, ikkunoiden järkevistä oletus- ja minimikokoarvoista sekä kokomuutoksista. Asettelunhallinta tekee päivityksiä sisältömuutosten tapahtuessa automaattisesti. Sisältömuutoksia ovat lapsikomponenttien kirjasimen tai muun sisällön koon muutos, lapsikomponentin piilottaminen tai näyttäminen ja lapsikomponenttien poistaminen.

Tässä opinnäytetyössä kehitetyssä analytiikassa tämän vaihtelevuuden vaikutusta kompensoidaan käyttämällä käyttöliittymäkomponentin koolla skaalattua hiiren kulkemaa matkaa. Skaalaaminen tehdään jokaiselle osoittimen siirtymäarvolle ennen kuin sen itseisarvo summataan askelessa kuljettuun kokonaismatkaan. Näin saadaan resoluutio- ja skaalariippumaton kulkumatka pikseliä kohti, joka on vertailukelpoinen esimerkiksi eri istuntojen välillä.

Laskenta nojaa oletukseen, että käyttöliittymäkomponentilta saadaan osoittimen siirtymän ilmoittava tapahtuma tavallisen päivitysaikavälin lisäksi aina komponentin rajalla, kun hiiren

osoitin on saapunut komponentin alueelle ja kun se on poistumassa komponentin alueelta. Normaalisti Qt päivittää hiiren sijaintitietoa käyttöliittymäkomponentille intensiivisesti vain, kun vähintään yksi hiiren näppäin on pohjassa. Työssä kehitetty analytiikka edellyttää, että tapahtumia saadaan jatkuvasti, joten hiiren osoittimen jatkuva päivittäminen (mouse tracking) aktivoidaan kyseiselle käyttöliittymäkomponentille hiiren osoittimen liikkuessa sen alueella.

Hiiren osoittimen skaalatun matkan (S) laskenta yhdelle askeleelle on kuvattu x-suunnassa alla (Kuvio 3 ja Kaava 1) ja se tehdään vastaavasti myös y-suunnassa pois lukien visualisoinnissa, jossa ne yhdistetään.



Kuvio 3. Hiiren osoittimen skaalatun matkan laskeminen

$$S_x = \sum_{k=0}^{N} \left| \frac{a_k}{W_i} \right| + \sum_{l=0}^{M} \left| \frac{b_l}{W_j} \right| + \ldots \qquad (1)$$

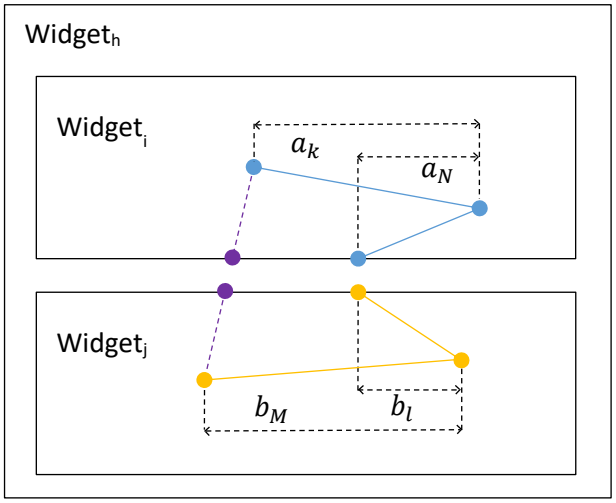
Mikäli Qt ei tuottaisi tapahtumia oletetun kaltaisesti (rajalla), olisi kyseessä vika, joka vaikuttaisi todennäköisesti muihinkin sovelluksiin. Mahdollinen vian aiheuttama virhe olisi selvitettävissä laskemalla komponenttirajan ylittävän siirtymän päätepisteiden etäisyydet rajasta ja näin saatujen arvojen suhteesta ylityskohta ja -aika, joihin tuotettaisiin kaksi ylimääräistä tapahtumaa. Asian selvittäminen on rajattu pois tämän opinnäytetyön piiristä.

6.4 Hiiren osoittimen kulkeman matkan hyötysuhde

Hiiren osoittimen kulkeman matkan hyötysuhdearvo kertoo missä suhteessa optimaalinen ja toteutunut matka ovat toisiinsa. Suuri hyötysuhdearvo voi esimerkiksi kertoa, että hiiren osoitinta on siirretty määrätietoisesti käyttäjäpolun askeleen lähtöpaikasta sen päättymispaikkaan.

Toisin kuin skaalatun matkan tapauksessa, hiiren osoittimen kulkeman matkan hyötysuhteen mittaamisessa käytetään pikselimäärää sellaisenaan. Lisäksi yksittäisten siirtymien arvot lasketaan yhteen *etumerkillisesti*, jota saadaan suora eli optimaalinen siirtymä askeleelle. Tämän siirtymän voisi laskea myös toteutuneiden loppu- ja alkupisteiden erotuksena, mutta tällöin mahdollisesti puuttuvat tapahtumat saattaisivat vääristää tuloksia. Puuttuvia tapahtumia ei kuitenkaan pitäisi olla, koska tapahtumasuodatin asennetaan sovellustasolle.

Optimaalisen siirtymän lisäksi askeleelle lasketaan toteutunut siirtymä summaamalla osamatkoja *itseisarvoja* yhteen. Hiiren osoittimen siirtymisen hyötysuhde (E_x) yhdessä askeleessa saadaan, kun jaetaan optimaalisen siirtymän matka (S_o) toteutuneen siirtymän matkalla (S). Asia on havainnollistettu alla olevassa kuviossa ja kaavassa (Kuvio 4 ja Kaava 2). Optimaalinen reitti on väritetty violetilla ja puuttuvia tapahtumia havainnollistetaan käyttöliittymäkomponentilla Widget_h, joka kuitenkin todellisuudessa tuottaisi tapahtumia mitä todennäköisimmin.



Kuvio 4. Hiiren osoittimen kulkeman matkan hyötysuhde

$$E_x = \frac{S_o}{S} = \frac{\sum_{k=0}^{N} a_k + \sum_{l=0}^{M} b_l + \ldots}{\sum_{k=0}^{N} |a_k| + \sum_{l=0}^{M} |b_l| + \ldots} \qquad (2)$$

Hyötysuhde asetetaan dimensiossa arvoon 1.0, kun toteutunut matka dimensiossa on nolla.

6.5 Poikkeamien tunnistaminen käyttäjäpolkujen askelissa

Poikkeava arvo on havainto, joka sijaitsee epänormaalin etäisyyden päässä muista arvoista populaation satunnaisotoksessa. Eräässä mielessä tämä määritelmä jättää analyytikon (tai konsensusprosessin) tehtäväksi päättää, mitä pidetään epänormaalina. Ennen kuin

epänormaalit havainnot voidaan erottaa, on tarpeen karakterisoida normaalit havainnot. (NIST 2024.)

Käyttäjäpolkujen poikkeamien tunnistus tehdään askeltasolla ja toteutetaan analytiikkapalvelimelta saatavaa pitkän aikavälin statistiikkaa vastaan z-score -menetelmällä. Menetelmä soveltuu hyvin, kun oletetaan että muuttujat ovat normaalijakautuneita (DataHeroes 2024). Jos askeleesta mitattu arvo tunnistetaan poikkeamaksi, se korvataan vastaavalla pitkän aikavälin keskiarvolla ja askeleen ko. muuttujan poikkeamalaskurin arvoa kasvatetaan. Poikkeamien tunnistusta sovelletaan sekä askeleen kestoon, hiiren osoittimen skaalattuun matkaan ja hiiren osoittimen matkan hyötysuhteeseen.

Askeleen poikkeamien tunnistusproseduuri lähtee liikkeelle, kun löydetään riittävän monta kertaa toistunut käyttäjäpolku. Tämän jälkeen etsitään pitkän aikavälin varastosta samanlainen käyttäjäpolku ja merkitään ylös myös mahdollinen vaihe-ero askelluksessa. Jos pitkän aikavälin varastosta löytyy samanlainen käyttäjäpolku, sitä vastaan lasketaan pisteytys tarkastelussa olevan käyttäjäpolun askelten mittausarvoille (Kaava 3). Varianssi muutetaan tätä varten keskihajonnaksi.

$$z = \frac{|x - \mu|}{\sigma} \qquad (3)$$

Jos z-arvo on suurempi kuin asetettu kynnysarvo, todetaan mittaustulos poikkeamaksi. Oletuskynnysarvo on 1.65 eli 95 % ja arvo 0 poistaa poikkeamien tunnistuksen käytöstä. Poikkeamien tunnistus tulisi ottaa pois käytöstä esimerkiksi silloin, kun analytiikkaa opetetaan, toisin sanoen, kun ensimmäistä pitkän aikavälin statistiikkaa kerätään.

6.6 Käyttäjäpolun tilastollisten tunnuslukujen käsittely

Yksittäisellä päätelaitteella kerättävän tapahtumajoukon aika-arvoista on mahdollista määrittää esimerkiksi minimi- ja maksimi-, keski-, keskihajonta-, ja varianssiarvoja. Käyttäjäpolkujen ja niiden askelten kestoista ja hiiren osoittimen kulkemista matkoista laskettavat tilastolliset tunnusluvut ovat oletettavasti hyödyllisiä tietoja laadullista käyttöliittymä- ja kokemustutkimusta tehtäessä. Käyttöliittymän tapahtumia ei ole tarve siirtää pois käyttäjän koneelta vaan riittää, että pelkästään tilastolliset tunnusluvut välitetään keskitetyn analytiikan palvelimelle.

Käyttäjäpolkujen tietojen käsitteleminen ja siirtäminen tilastollisina tunnuslukuina parantaa operatiivisen tiedon tietosuojaa, kun samalla varmistetaan minimiotoskoon toteutuminen ennen arvojen siirtämistä pois päätelaitteelta. Näiden lukujen yhdistäminen on mahdollista

keskitetysti, kun otoskoot ja keskiarvojen varianssi välitetään jokaisen tilastollisen aikatiedon liitteenä. Tilastollisten tunnuslukujen käsitteleminen ja siirtäminen raa'an tapahtumadatan sijaan saattaa myös olla tehokkaampaa, mikä pienentää analytiikkatoiminnallisuuden koko järjestelmälle tulevaa lisäkuormaa. Analytiikkatoiminnallisuuden ei ole suotavaa aiheuttaa havaittavaa muutosta järjestelmän suorituskykyyn.

6.7 Käyttäjäpolkujen automaattinen tunnistaminen

Käyttäjäpolkujen automaattinen tunnistaminen tehdään ohjelman suorituksen lopussa käyttäen istunnon aikana tallennettua tapahtumavektoria. Analytiikkalisäosan parametreina saadaan askeleen maksimipituus, käyttäjäpolkuinstanssien minimilukumäärä, käyttäjäpolun minimiaskelmäärä ja käyttäjäpolun maksimiaskelmäärä. Tunnistusalgoritmi on kuvattu alla.

Tunnistaminen lähtee liikkeelle käyttämällä polun maksimiaskelmäärää, jonka mittaista ikkunaa liu'utetaan tapahtumavektorin yli. Tätä liukuvaa ikkunaa kutsutaan tässä osumatestin lähdevektoriksi. Tämän vektorin ominaisuuksia verrataan toisen, saman pituisen ja niin ikään tapahtumavektorin yli liu'utettavan kohdevektorin ominaisuuksiin.

Vektorien tapahtumien pitää täsmätä askeltunnisteiden osalta, eikä yksikään kohdevektorin tapahtuma saa olla merkitty käytetyksi. Lisäksi muilta kuin ensimmäiseltä tapahtumalta vaaditaan, että sen kesto on lyhyempi kuin askeleen maksimipituus, jolloin käyttäjäpolun keskelle ei tule tahattoman pitkiä askelia. Askeltunnisteet ovat merkkijonoja, jotka koostuvat kyseisen tapahtuman tuottaneen käyttöliittymäkomponentin yksiselitteisestä tunnisteesta, tapahtumatyypistä ja mahdollisista näppäintiedoista.

Kun kaikki tapahtumat on käyty läpi ristiin sekä lähdevektorin ikkunan että kohdevektorin ikkunan osalta, katsotaan, onko täsmääviä käyttäjäpolkuja löytynyt vaaditun käyttäjäpolkuinstanssien minimilukumäärän verran. Jos uusi käyttäjäpolku on löytynyt, kaikki niihin käytetyt tapahtumat merkitään käytetyiksi. Käyttäjäpolulle luodaan tunniste, joka sisältää muun muassa tapahtuman tuottaneen käyttöliittymäkomponentin ja sen vanhempien tunnisteet.

Tämän jälkeen suoritetaan mittausarvoille poikkeamien tunnistus ja huomiointi sekä askelille tilastollisten tunnuslukujen laskenta. Käyttäjäpolun tallentamisen jälkeen siirrytään seuraavaksi pienempään askelmäärään. Tätä jatketaan, kunnes alitetaan käyttäjäpolun minimiaskelmäärä. Tarvittava tapahtuma vs. tapahtuma -testien lukumäärän N_t voi laskea kaavalla 4, jossa s_{min} on käyttäjäpolun minimiaskelmäärä, s_{max} on käyttäjäpolun maksimiaskelmäärä ja N_e on tapahtumavektorin pituus. Algoritmin aikakompleksisuus on tällöin $O(S_{max} * N_e^2)$.

$$N_t = \sum_{n_s=s_{max}}^{s_{min}} n_s * (N_e - n_s + 1)^2 \qquad (4)$$

6.8 Lämpökarttojen yhdistäminen

Käyttöliittymäkomponentin kahden erillisen lämpökartan yhdistäminen tapahtuu aina pareittain analytiikkapalvelimella. Käyttöliittymäkomponentin lämpökartan ohessa on aina liitettynä komponentin täydellinen tunniste, jolloin toisiaan vastaavat kuvat ovat löydettävissä eri istuntojen ja jopa eri ohjelmistoversioiden tuottamien tallenteiden välillä. Jos kuvien resoluutiot ovat eri suuruiset, interpoloidaan niistä ennen yhdistämistä uudet kuvat, joiden kumpaankin dimensioon molemmat alkuperäiset kuvat mahtuvat.

Varsinainen yhdistäminen suoritetaan pikseleittäin, jolloin huomioidaan molempien kuvien pikselien väriarvo (myös alpha) vastaavissa kohdissa ja painotetaan niitä kyseisten datojen keräysajoilla. Istunnon datan painoarvo on istunnon datan keräysaika jaettuna istunnon datan keräysajan ja pitkän aikavälin datan keräysajan summalla. Pitkän aikavälin datan painoarvo on pitkän aikavälin datan keräysaika jaettuna istunnon datan keräysajan ja pitkän aikavälin datan keräysajan summalla.

Värikomponenttien kumuloituvat arvot lasketaan yhdistämisessä 64-bittisinä liukulukuina, mutta tallennetaan 32-bittisinä liukulukuina. Saturoituminen ei ole todennäköistä, kunhan käytetään maltillista lämpökarttojen kerrytysnopeutta. Lämpökarttojen ohessa säilytetään myös kyseisiin käyttöliittymäkomponentteihin osuneiden hiirivalintojen lukumääriä. Nämä lukumäärät yhdistetään laskemalla ne yhteen.

6.9 Käyttäjäpoluista laskettavat metriikat

Käyttäjäpolkujen askelille lasketaan tilastolliset tunnusluvut niiden tunnistamisen ja poikkeamien huomioimisen jälkeen. Itse käyttäjäpoluille tilastolliset tunnusluvut lasketaan vasta juuri ennen visualisointia, eikä niitä tallenneta tietokantoihin. Tämä voidaan tehdä, koska kaikki käyttäjäpolun tilastolliset tunnusluvut ovat määritettävissä sen lasten tilastollisten tunnuslukujen perusteella.

Tilastollisia tunnuslukuja lasketaan käyttäjäpolun askeleen kestolle, hiiren osoittimen askeleessa kulkemalle skaalatulle matkalle sekä hiiren osoittimen kulkeman matkan hyötysuhde askeleessa. Ensimmäistä lukuun ottamatta tunnusluvut lasketaan ja tallennetaan x- ja y-suunnille erikseen. Tunnuslukuja ovat kunkin osalta minimi-, maksimi-, ja keskiarvot sekä varianssi. Mediaania ja kvartiileja ei lasketa, koska niitä ei datapisteiden puuttuessa

kuitenkaan voisi yhdistää tallennettuihin pitkän aikavälin tilastotietoihin. Näppäimistötyön määrän mittaamiselle ei nähty tarvetta.

6.10 Käyttäjäpolkujen yhdistäminen

Käyttäjäpolkujen yhdistäminen tapahtuu myös analytiikkapalvelimella kahden käyttäjäpolkujoukon välillä. Jokaiselle istunnon käyttäjäpolulle etsitään vastine pitkän aikavälin käyttäjäpolkujen joukosta ja jos vastine löytyy, siirrytään yhdistämään kyseisiä polkuja. Jos vastinetta ei löydy, istunnon käyttäjäpolku siirretään sellaisenaan pitkän aikavälin polkujen joukkoon. Vastaavuustesti huomioi luonnollisesti myös käyttäjäpolkujen askelvaiheiden eroavaisuuden.

Käyttäjäpolkujen yhdistämisessä yhdistetään niiden askeleet ja instanssilaskurit. Käyttäjäpolkujen yhdistämisessä yhdistetään niiden kaikki askeleet toisiinsa huomioiden mahdollinen ero askelvaiheissa. Askelten yhdistettyjen minimi- ja maksimiarvojen laskeminen on suoraviivaista, mutta yhdistäminen keskiarvojen (Kaava 5) ja varianssien (Kaava 6, BBSysDyn 2022; Rudmin 2010) osalta tehdään tässä opinnäytetyössä käyttämällä seuraavia kaavoja:

$$\mu = \frac{\mu_1 \ * \ n_1 + \mu_2 \ * \ n_2 + \dots}{n_1 + n_2 + \dots} \tag{5}$$

$$\sigma^2 = \frac{n_1 {\sigma_1}^2 + n_2 {\sigma_2}^2 + \dots}{n_1 + n_2 + \dots} + \frac{n_1 (\mu_1 - \mu)^2 + n_2 (\mu_2 - \mu)^2 + \dots}{n_1 + n_2 + \dots} \tag{6}$$

Askelten välillä yhdistetään myös metriikkakohtaiset poikkeamalaskurit.

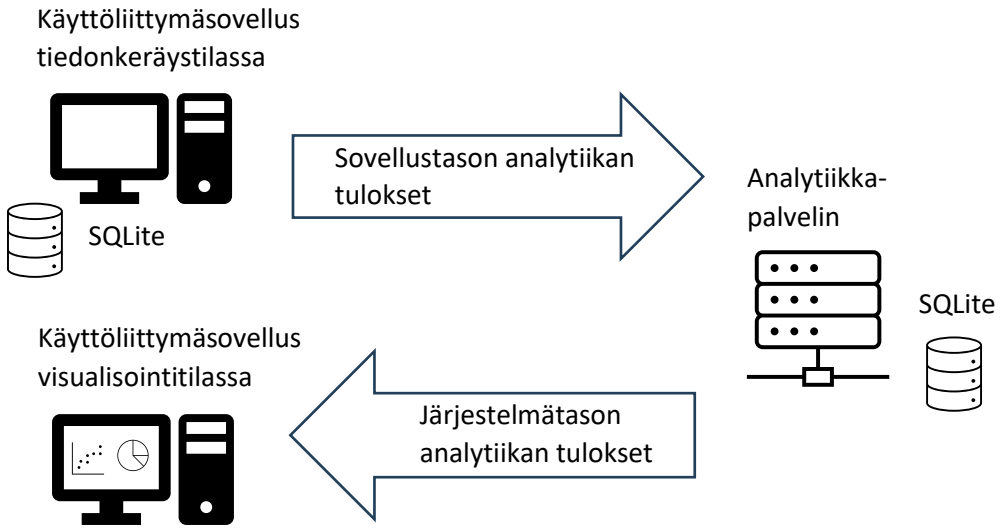
7 Käyttöanalytiikkajärjestelmä

7.1 Yleistä

Erillistä pilvi-infrastruktuuria on vaikeaa saada asennettua asiakkaidemme turvakriittisiin erillisverkkoihin ja julkisen pilven käyttö on periaatteessa poissuljettu vaihtoehto. Tarvittava toiminnallisuus voidaan kuitenkin sisällyttää tuotteen käyttöliittymäsovellukseen ja kehitettävään palvelinsovellukseen. Analytiikkalisäosalla varustettuna mikä tahansa analysoitava Qt-pohjainen käyttöliittymäsovellus toimii sekä datan keruualustana, että kerätyn datan ja data-analytiikan tulosten analysointialustana.

Arkkitehtuuri

Alla olevassa kuvassa on esitetty uuden käyttöanalytiikan järjestelmäarkkitehtuuri.



Kuva 1. Käyttöanalytiikan järjestelmäarkkitehtuuri

Asiakastyöasemalla suoritettava käyttöliittymäsovellus, joka on tiedonkeruutilassa, lähettää tilastolliset tulostiedot analytiikkapalvelimelle säännöllisesti ja sovellusta suljettaessa. Analysointityöasemalla ajettava käyttöliittymäsovellus, joka puolestaan on käynnistetty tulosten esitystilaan, kysyy sovelluksen käynnistyessä ja sen jälkeen säännöllisesti uusimmat tilastolliset tulostiedot analytiikkapalvelimelta.

Asiakas-palvelin (client-server) -arkkitehtuurin sijaan työssä harkittiin myös ilman palvelinta toimivaa, vertaiskommunikaatioon (P2P) perustuvaa järjestelmää. Keskitetty pitkän aikavälin tietokanta kuitenkin helpottaa järjestelmän hallintaa ja valvontaa verrattuna vertaiskommunikaatioon, jossa jokaisella käyttöliittymäsovelluksella analytiikkalisäosineen on sama

asema ja vastuu. Lisäksi käyttöliittymäsovellukset käynnistetään ja sammutetaan yksittäisen käyttäjän tarpeen mukaan, kun palvelinsovellus voi olla käytännössä aina käynnissä. Pitkän aikavälin tietokanta keskitetyllä palvelimella on myös helpommin varmuuskopioitavissa ja siirrettävissä järjestelmien välillä.

Toteutus on alustariippumaton ja on siten käännettävissä esimerkiksi Windowsissa, Linuxissa ja Mac OS:ssä. Varsinaisia laitteistovaatimuksia ei asiakastyöasemalle ja palvelimelle ole Qt:n asettamien laitteistovaatimusten lisäksi. Asiakastyöaseman ja palvelimen välillä on kuitenkin oltava mahdollisuus muodostaa TCP/IP-yhteys.

Skaalautuvuus

Järjestelmän säilöessä tietoa lämpökarttoina ja tilastollisina tunnuslukuina sen skaalautuvuus on parempi kuin tapahtumia tallentavan järjestelmän, koska datan määrä ei kasva merkittävästi ajan kuluessa ja käyttöliittymäsovellusten lukumäärän kasvaessa. Toteutettu järjestelmä ei siis rajoita käyttöliittymäsovellusten lukumäärää eikä keruujakson pituutta. Datan keruu, varastointi ja visualisointi puolestaan skaalautuvat hyvin erikokoisille näyttöresoluutioille.

Ominaisuudet

Qt:ssa tapahtumasuodattimen (eventfilter) avulla on mahdollista havainnoida käyttöliittymäkomponenttien tuottamia tapahtumia (QEvent). Tämän voi asentaa mille tahansa käyttöliittymäkomponentille tai itse sovellusinstanssille. Tapahtumasuodatin asennettiin sovellusinstanssille, jolloin analytiikkatoiminnallisuuden ulottuville saadaan kaikki sovelluksen käyttöliittymässä tuotettavat tapahtumat. Tämän ansiosta käyttöliittymän rakennetta ja tilaa voidaan jäljittää ja mitata kattavasti. Tarvittaessa sovellus voi tuottaa myös kustomoituja tapahtumia, joilla saadaan muutkin kuin Qt:n hallinnoimien kontrollien tapahtumat talteen.

Tässä opinnäytetyössä toteutetulla analytiikalla on käytössään samat tapahtumatiedot kuin esimerkiksi Qt Insight -tuotteella. Tässä kappaleessa kuvattu toteutus on kuitenkin opinnäytetyössä tehdyn kehittämistyön tuotosta, eikä esimerkiksi Qt Insight -tuotteesta lainattua. Qt:n toimintoja on muuten käytetty laajasti hyväksi esimerkiksi kommunikaatiossa verkon yli, tietokantojen käsittelyssä ja tilastollisten tunnuslukujen visualisoinnissa.

Analytiikkalisäosan ja -palvelimen muodostaman järjestelmän perusominaisuuksiin kuuluu

- käyttöanalyysien tekeminen ja raporttien tuottaminen niistä
- toiminta suljetussa ja rajoitetussa ympäristössä
- analytiikan lisättävyys käyttöliittymäohjelmistoon Qt-lisäosana ilman tarvetta ”build-data” sovellus uudestaan

- tapahtumadatan keräämisen tehokkuus siten, ettei normaali käyttö häiriinny
- paikallisten tulosten varastointi asiakastyöasemalla palvelinyhteyden saamiseen asti
- tulosten yhdistäminen ja varastointi palvelimella
- kertaalleen analyysiin käytetyn datan uudelleenkäyttämättömyys muuten kuin poikkeamien (outliers) tunnistamiseen käyttäjäpoluissa
- sovelluksen käynnistettävyys keräystilaan, visualisointitilaan tai demonstraatiotilaan
- datan analysointi monivaiheisesti käyttöliittymäsovelluksen sammuessa, analytiikkapalvelimella tarpeen mukaan ja käyttöliittymäsovelluksessa juuri ennen visualisointia
- tulosten visualisointi toimivan käyttöliittymän päällä visualisointitilassa ja demonstraatiotilassa.

Analytiikka on lisättävissä käyttöliittymäohjelmistoon Qt-lisäosana (plugin), jolloin se on helppo ottaa mukaan sovellukseen ja jättää siitä pois. Tästä syystä analytiikka on liitettävissä mihin tahansa Qt-pohjaiseen sovellukseen. Kertaalleen analyysiin käytetyn datan päätyminen uudestaan analyysiin estetään sillä, että analytiikkalisäosa lähettää palvelimelle tuloksia vain keräystilassa.

On huomioitavaa, että analytiikan ja Qt-käyttöliittymäkomponenttien nimeämisen kattavuudet kulkevat käsi kädessä, eli mitä suurempi osuus käyttöliittymäkomponenteista on nimetty, sitä todenmukaisempia analytiikan tuloksetkin ovat. Käyttöliittymäkomponenttien kattava nimeäminen on perusedellytys myös käyttöliittymän tyylien ja automaattitestauksen toiminnalle ja laajalle kattavuudelle. Nimeämättömiä käyttöliittymäkomponentteja voi kuitenkin olla pitkän elinkaaren sovelluksessa runsaasti, joten niiden määrittämiseen saattaa olla tarpeen varata huomattava määrä aikaa.

Raportointi ei paljasta yksittäisen käyttäjän tietoja tai asiakkaan sensitiivistä dataa, koska raportoinnissa pitäydytään tilastotiedoissa. Tulokset ovat kokonaisuudessaan asiakkaan tarkastettavissa ennen kuin niitä viedään pois asiakkaan järjestelmästä. Analytiikkajärjestelmä tuottaa asiakkaalle perustoiminnallisuutena raportteja tunnistettujen käyttäjäpolkujen tilastollisista tunnusluvuista taulukkomuodossa, käyttäjäpolkujen askelten tilastollisista tunnusluvuista taulukkomuodossa ja graafisena, hiiren osoittimen liikkeen ja klikkausten tuottamana lämpökarttoina, hiiren klikkauskertojen laskureina käyttöliittymäkomponenteittain, käyttäjäpolkujen askelten järjestysnumeroina käyttöliittymäkomponenteittain ja käyttöliittymän rakennetta käytetyltä osin tekstimuodossa konsoliin tai tiedostoon. Käyttäjäpolkujen

tunnistaminen huomioi myös pikanäppäinten ja lisänäppäinten (modifiers) käytön. Lämpökartat interpoloidaan kulloisiinkin käyttöliittymäkomponenttien kokoihin eli resoluutioihin ja esitetään käyttöliittymän päällä demo- ja visualisointimoodeissa.

7.2 Tapahtumadata ja sen käyttäminen

Tapahtumapohjaisessa ohjelmointimallissa tapahtumat määräävät ohjelman kulun. Tapahtumia ovat käyttäjän toimet, esimerkiksi hiiren siirtyminen tai näppäimistöpainallukset, sekä järjestelmäilmoitukset ja muut tapahtumat. Yksinkertainen yritys määrittää tapahtumaohjattu ohjelmointi (EDP) voisi Lukkarisen (2022, 30) mukaan nojata sen keskeisten käsitteiden määrittelyyn. Tällaisia käsitteitä voisivat hänen mielestään olla muun muassa tapahtuma, tapahtumakäsittelijä, tapahtumakäsittelijän lisääminen ja poistaminen, tapahtumajono ja tapahtumasilmukka.

Qt tarjoaa vakaan tapahtumajärjestelmän, jonka avulla kehittäjä voi käsitellä näitä tapahtumia ja vastata niihin ja tuottaa kaikki graafisen käyttöliittymän tilan ja toiminnallisuuden vaatimat tapahtumat. Tapahtumien havainnointi tapahtuu Qt-sovellukselle asennetun tapahtumasuodattimen (EventFilter) avulla. Sovelluksen analytiikkalisäosa käsittelee näitä tapahtumia kaikissa toimintatiloissa, mutta varsinaista hyötydataa kerätään tapahtumista vain keräys- ja demonstraatiotiloissa. Varsinainen hyötydata koostuu KeyPress, MouseMove ja MouseButtonRelease -tapahtumista, mutta tärkeää osaa esittävät myös muut, datan tallennushierarkiaan ja visualisointiin vaikuttavat tapahtumat.

Kerättävät tapahtumat ja niiden käyttötarkoitukset kohdesovelluksessa on lueteltu taulukossa 3. Käyttöliittymän rakenne rekonstruoidaan varjomallirakenteeseen siltä osin kuin sitä käytetään. Jokaista tapahtumia tuottavaa käyttöliittymäkomponenttia kohden luodaan sen tilaa seuraava varjomalli.

Tapahtumatyyppi	Käyttötarkoitus analytiikkalisäosassa
`ChildAdded`	Lisää varjomallille olemassa olevan lapsen. Luo tarvittaessa uuden varjomallin lapseksi lisäten vikasietoisuutta kadonneen tapahtuman varalta. Tarkistetaan ja päivitetään tarvittaessa ylätason varjomallijoukko siltä varalta, että lapsi ei tuottanut `ParentChange` tapahtumaa.
`ChildRemoved`	Poistaa varjomallilta lapsen. Tarkistetaan ja päivitetään tarvittaessa ylätason varjomallijoukko siltä varalta, että lapsi ei tuottanut `ParentChange` tapahtumaa.
`Create`	Luodaan varjomalli. Kytkeytyminen vanhempimalliin on mahdollinen samalla. Tarkistetaan ja päivitetään tarvittaessa ylätason varjomallijoukko. Läpinäkyvän visualisointikerroksen piirtäminen lähtee liikkeelle näistä ylätason varjomalleista. Luo tarvittaessa uuden varjomallin vanhemmaksi lisäten vikasietoisuutta kadonneen tapahtuman varalta.
`Destroy`	Tuhotaan varjomalli. Poistetaan mahdolliset sidokset vanhempimalliin ja lapsimalleihin.
`Enter`	Kytkee hiiren osoittimen jäljityksen päälle ko. käyttöliittymäkomponentille. Vaikuttaa myös lämpökarttaa keräävän varjomallin valintaan.
`FocusIn`	Mahdollistaa fokusajan keräämisen aloittamisen ko. varjomallille.
`FocusOut`	Mahdollistaa fokusajan keräämisen lopettamisen ko. varjomallilta.
`Hide`	Ohjaa varjomallin näkyvyystilaa
`KeyPress`	Muodostaa käyttäjätapahtuman, johon sisällytetään aikaleima käyttäjäpolkujen tunnistamista ja askelviiveen laskemista varten.
`Leave`	Kytkee hiiren osoittimen jäljityksen pois päältä ko. käyttöliittymäkomponentilta. Vaikuttaa myös lämpökarttaa keräävän varjomallin valintaan.
`MouseButtonRelease`	Muodostaa käyttäjätapahtuman, johon sisällytetään aikaleima käyttäjäpolkujen tunnistamista ja askelviiveen laskemista varten. Käytetään täten vain, jos käyttöliittymäkomponentti on näkyvissä. Lisäksi tämä tapahtuma määrittää hiiren osoittimen suoran ja epäsuoran siirtymän mittaamista.
`MouseMove`	Jos käyttöliittymäkomponentti on näkyvissä, määrittää hiiren osoittimen suoran ja epäsuoran siirtymän mittaamista.
`ParentChange`	Asettaa varjomallille uuden vanhempimallin. Luo tarvittaessa uuden varjomallin vanhemmaksi lisäten vikasietoisuutta kadonneen tapahtuman varalta. Tarkistetaan ja päivitetään tarvittaessa ylätason varjomallijoukko.
`Resize`	Muuttaa varjomallissa tallennettavan lämpökartan kokoa, josta käyttöliittymäkomponentin koko on kysyttävissä myöhemminkin.
`Show`	Ohjaa varjomallin näkyvyystilaa

Taulukko 3. Tapahtumatyypit ja niiden käyttötarkoitus analytiikkalisäosassa

KeyPress, MouseMove ja MouseButtonRelease -tapahtumista luodaan istunnon ajalta käyttäjätapahtumavektori, jonka alkiot sisältävät muun muassa keston edelliseen käyttäjätapahtumaan, viittaus liittyvään varjomalliin, käyttäjäpolun askeleen tunnisteen ja viimeisen käyttäjätapahtuman jälkeen toteutuneen hiiren kulumatka. Lisäksi käyttäjätapahtuma sisältää kloonatun Qt-tapahtuman, jolloin se on käytettävissä erillisessä, tapahtumienkäsittelyä suorittavassa, säikeessä.

7.3 Käyttöliittymäsovelluksen analytiikkalisäosa

Käyttöliittymän toimintaa ja käyttöä seuraava analytiikkalisäosa on Qt:n matalan tason sovellusohjelmointirajapintaa (API) käyttävä ohjelmistomoduuli, joka voidaan ladata ajon aikana mihin tahansa Qt:n päälle rakennettuun sovellukseen. Tämän edellytyksenä on kuitenkin se, että sovellus lataa lisäosan, vaikka ei tunne sen tyyppiä. Analytiikkalisäosa ei toteuta mitään rajapintaa, mitä itse analysoitava sovellus voisi käyttää.

Analytiikkalisäosan päätehtävinä on kerätä analysoitavan sovelluksen käyttötapahtumadataa istunnon ajalta, luoda käyttötapahtumadatasta käyttäjäpolut tilastollisine tunnuslukuineen, visualisoida käyttäjäpolut askelineen taulukkomuodossa ja suoritettavan käyttöliittymän päällä, kerätä lämpökartat hiiren liikkeestä ja hiiren valinnoista sekä visualisoida lämpökartat ja valintalukumäärät niin ikään suoritettavan käyttöliittymän päällä. Analytiikkalisäosa on hyvä ladata ensimmäiseksi, jotta se saa varmasti kaikki istunnon tapahtumat talteen. Tämä on tärkeää erityisesti siksi, että analytiikkalisäosan toiminta perustuu hyvin pitkälti varjomallirakenteeseen, joka rakennetaan ja pidetään yllä tiettyjen perustyyppisten tapahtumien avulla. Tapahtumien käsittelyssä ja varjomallirakenteen toteutuksessa on kuitenkin puuttuvien tapahtumien varalta vikasietoisuutta, joka näkyy myös sivun 34 taulukossa.

Käyttöliittymäkomponenttien uniikki nimeäminen vähintään sisarkomponenttien kesken on tärkeää, jotta niiden datat pysyvät erillään. Tunnisteet varjomalleille luodaan käyttämällä sen komponentin omaa objektinimeä ja kaikkien vanhempikomponenttien objektinimiä aina juurikomponenttiin asti. Käyttöliittymäkomponentin muistiosoite toimii istunnon aikana tehokkaana tunnisteena sidoksissa oleviin muihin varjomalleihin, mutta osoitteen datasisältöä ei kuitenkaan Qt:n määrityksen mukaan saa käyttää muualta kuin pääsäikeestä. Käyttöliittymäkomponenttien nimi puolestaan toimii universaalina tunnisteena, joka on validi myös istuntojen välillä.

Nimeämättömien käyttöliittymäkomponenttien yksiselitteisyys olisi mahdollista varmistaa lisäämällä jokaisen käyttöliittymäkomponentin nimen perään lapsijärjestysnumeron (esimerkiksi "#3"), mutta järjestysnumerojen hallinnointi reaaliajassa dynaamisen käyttöliittymän

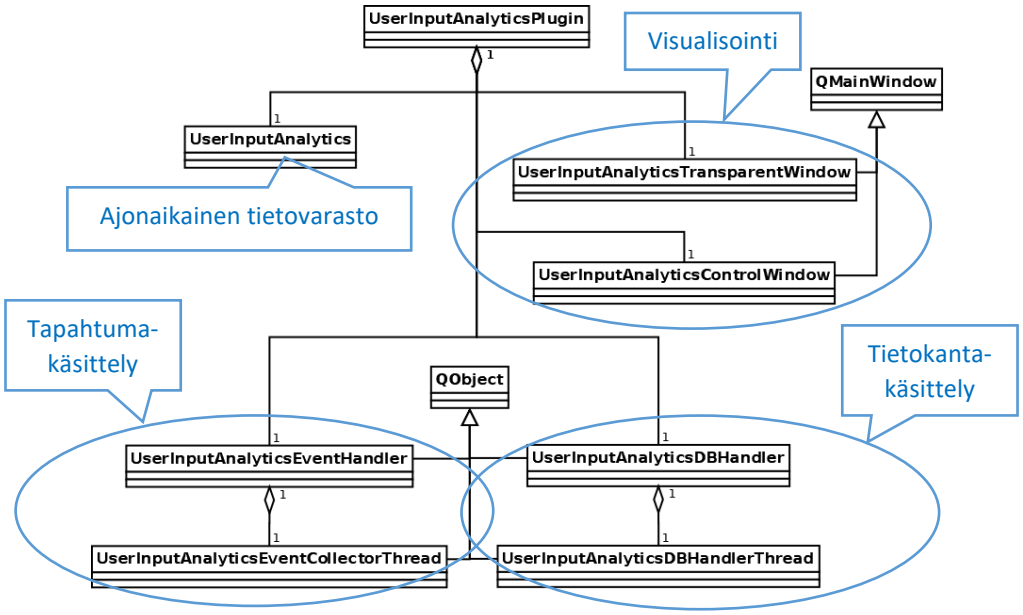
muutosten mukaan saattaisi tulla liian raskaaksi. Nyt varjomalli säilyttää lapsiaan järjestämättömässä joukko-oliossa, jonka käsittely on varsin tehokasta suurillakin joukoilla.

Toiminnan ei tule hidastaa normaalia käyttöä havaittavasti eikä aktiivisuuden pidä näkyä käyttäjälle keräystilassa. Koska käyttötapahtumia keräävä tapahtumasuodatin on asennettu sovellustasolle, on se kuitenkin potentiaalisesti kuormittava. Tämän kuormituksen hajauttamisessa pääsäikeessä suoritettava tapahtumien esikäsittelijä laittaa analytiikan tarvitsemat tiedot jonoon, josta niitä puretaan säikeistetysti. Qt:n määritelmän mukaan käyttöliittymäkomponentteja ei saa kuitenkaan käyttää muusta kuin pääsäikeestä, joten kaikki analyysin tarvitsemat tiedot on kerättävä talteen jo tapahtumien esikäsittelijässä.

Datan keräystilassa tapahtuvassa lämpökarttojen kerrytyksessä vain päällimmäinen käyttöliittymäkomponentti kerryttää lämpökarttaa. Tämä on tärkeää, jotta kerrytys ei tapahdu moneen kertaan samassa kohdassa. Lämpökarttojen ohessa jokaisen käyttöliittymäkomponentin osalta pidetään kirjaa siihen osuneiden hiirivalintojen lukumääristä.

Kuvio 5 esittää käyttöliittymäsovelluksen analytiikkalisäosan rakenteen karkealla tasolla. UserInputAnalyticsPlugin eli sovelluslisäosan pääluokka luo lisäosan kaikki käsittelijä- ja ikkunapääobjektit ja asentaa tapahtumasuodattimen sovellukseen. Analytiikan tulosten sekä visuaalisesti että käyttäjän syötteelle läpinäkyvä ja koko ruudun peittävä visualisointiikkuna luodaan vain visualisointi- ja demotiloissa.

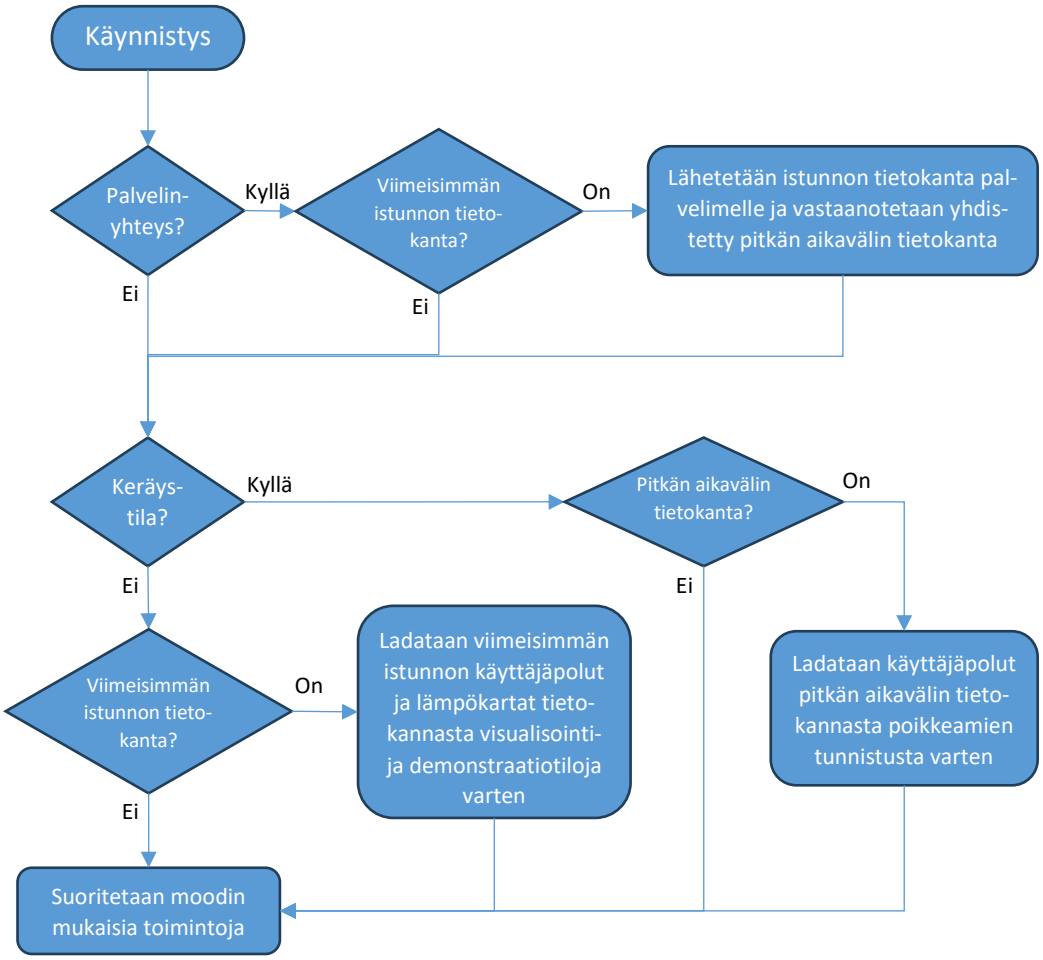
UserInputAnalytics-luokan instanssi sisältää yhteiset muuttujat ja tietovarastot, joten se jaetaan kaikille pääobjekteille. Pääobjektit käyttävät sitä keskinäiseen kommunikointiin säieturvallisesti. Lisäksi jotkut pääobjektit kommunikoivat suoraan toistensa kanssa, kuten pääsäikeessä suoritettava tapahtumakäsittelijä (EventHandler), joka syöttää jalostettuja tapahtumia suoraan tapahtumienkerääjäsäikeelle (EventCollectorThread). Käyttöliittymä muodostuu useista itse tehdyistä luokista, sisältäen lukuisia attribuutteja ja toiminnallisuuksia, joista Kuvio 5 selkeyden vuoksi kuvaa ainoastaan nimet ja hierarkian.



Kuvio 5. Käyttöliittymäsovelluksen analytiikkalisäosan rakenne

7.4 Käyttöliittymäsovelluksen analytiikkalisäosan toimintatilat

Käyttöliittymäsovellukselle annettavat käynnistysparametrit välittyvät myös analytiikkalisäosalle ja se voidaan käynnistää kokonaan ilman analytiikkatoiminnallisuutta, tiedonkeräystilassa, tulostenesitystilassa tai demonstraatiotilassa. Käytettävä tila valitaan käynnistysparametrilla viitatussa asetustiedostossa olevan parametrin avulla. Analytiikkalisäosa lukee asetustiedoston käynnistyksen yhteydessä ja päättää sen perusteella mihin tilaan se aktivoi itsensä. Toimintatilojen merkitys käynnistyksessä on esitetty kuviossa 6.



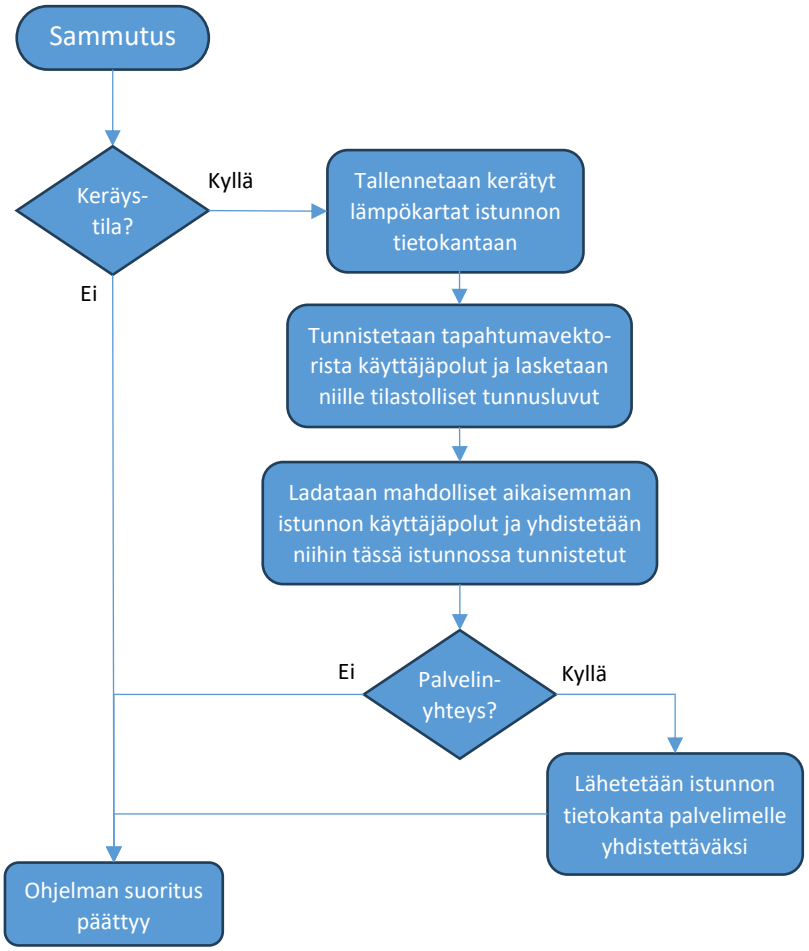
Kuvio 6. Toimintatilojen merkitys käynnistyksessä

Visualisointi- ja demonstraatiotiloissa analytiikkalisäosa luo kaksi erillistä pääikkunaa. Toinen on läpinäkyvä lämpökarttojen, klikkauslukumäärien ja käyttäjäpolkujen askelten järjestysnumeroiden esittämistä varten ja toinen tavallinen ikkuna tilastollisten tunnuslukujen esittämistä ja visualisointivalintojen tekemistä varten. Läpinäkyvä ikkuna peittää aina koko näyttöruudun, mutta päästää käyttäjän valinnat läpi vaikuttamatta niihin.

Eri moodeille yhteinen toiminnallisuus kattaa sekä palvelinyhteyden että viimeisimmän istunnon tietokannan olemassaolon tarkistamisen. Jos molemmat ovat olemassa, viimeisimmän istunnon tietokanta lähetetään palvelimelle ja vastauksena saadaan palvelimen yhdistämä pitkän aikavälin tietokanta. Vaikka palvelin vastaanottaisi tyhjän tietokannan, se kuitenkin lähettää pitkän aikavälin tietokannan takaisin. Tällöin mitään yhdistettävää ei ole, mutta tyhjä tietokanta toimii kuitenkin kyselynä. Analytiikkalisäosa poistaa paikallisen tietokannan aina vastauksen saatuaan.

Tiedonkeräystilassa analytiikkalisäosa ei vastaanota mitään tilastollisia tunnuslukuja laskennan pohjaksi, etteivät tulokset lähde kiertämään. Palvelimelta saatavasta pitkän aikavälin tietokannasta ladataan käyttäjäpolut ainoastaan poikkeamien tunnistuksen vertailukäyttöön. Käynnistettäessä ilman palvelinyhteyttä analytiikkalisäosa lukee tilastolliset tunnusluvut paikallisesta tietokannasta muistiin. Jos tiedonkeräystilassa ei ole saatavilla pitkän aikavälin tietokantaa, poikkeamien tunnistusta ei tehdä. *Visualisointitilassa* käynnistettäessä analytiikkalisäosa lataa pitkän aikavälin tai viimeisimmän istunnon tietokannan sisällön visualisointia varten. Palvelimelle päin ei visualisointitilassa lähetetä mitään. *Demonstraatiotilassa* lämpökartat kerätään normaalia nopeammin ja näytetään reaaliajassa. Käyttäjäpolut tunnistetaan pyydettäessä eli painamalla painiketta, eikä tällöin huomioida poikkeamia. Tilastollisten tunnusluvut lasketaan vain kyseisessä istunnossa kertyneiden tapahtumien osalta. Demonstraatiotilassa ei käytetä palvelinyhteyttä eikä tuloksia talleteta tietokantoihin. Demotila on hyödyllinen myös paikallisen keräystoiminnon ja visualisoinnin vianetsinnässä.

Sammutuksessa erityistä toiminnallisuutta on käytännössä vain, kun ollaan keräystilassa. Toimintatilojen merkitys sammutuksessa on esitetty kuvion 7 avulla. Tällöin istunnon tietokannassa mahdollisesti ennestään olevat lämpökartat korvataan meneillään olevan istunnon aikana kerätyillä lämpökartoilla, jonka jälkeen tunnistetaan istunnon aikana kerätystä tapahtumavektorista käyttäjäpolut ja lasketaan niille tilastolliset tunnusluvut. Tunnistetut käyttäjäpolut yhdistetään tilastollisine tunnuslukuineen mahdollisesti aikaisemman istunnon käyttäjäpolkuihin (kuten palvelimellakin) ja tallennetaan istunnon tietokantaan. Lopuksi istunnon tietokanta lähetetään palvelimelle yhdistettäväksi pitkän aikavälin tietokantaan, mutta vastausta ei jäädä odottelemaan, vaan päätetään suoritus.



Kuvio 7. Toimintatilojen merkitys sammutuksessa

7.5 Analytiikkapalvelin

Analytiikkapalvelin on itsenäinen sovellus, jonka tehtävä on ylläpitää pitkän aikavälin tietokantaa tilastollisista tunnusluvuista ja yhdistää siihen eri käyttöliittymäsovellusten analytiikkalisäosilta saatavia istuntojen tietokantoja. Lisäksi analytiikkapalvelimen tehtävänä on palauttaa yhdistetty pitkän aikavälin tietokanta analytiikkalisäosalle aina saatuaan istunnon tietokannan yhdistettyä. Analytiikkalisäosa voi käytännössä pyytää pitkän aikavälin tietokannan lähettämällä tyhjän tietokannan palvelimelle, jolloin yhdistämistä ei tapahdu, mutta se saa kuitenkin vastauksena pitkän aikavälin tietokannan, jota se voi käyttää esimerkiksi poikkeamien tunnistukseen.

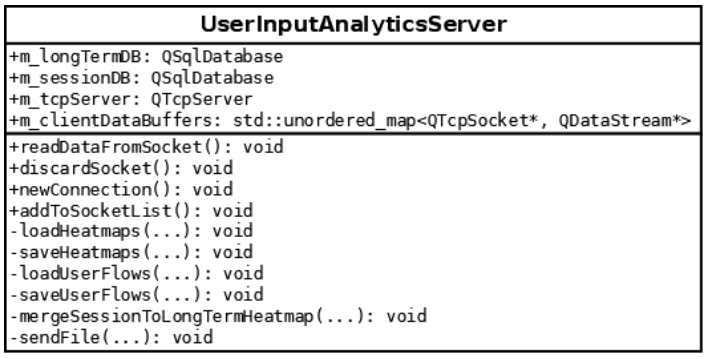
Palvelin tukee useita samanaikaisia yhteyksiä eri käyttöliittymäsovelluksiin, mutta käyttöliittymäsovellukselta tulevan istunnon tietokannan ja pitkän aikavälin tietokannan yhdistäminen tapahtuu aina atomisena operaationa. Pyynnöt käsitellään peräjälkeen yksisäikeisessä käsittelijässä, jonne pääsyä pyynnöt odottavat TCP-puskurissa. Paketin eheyden

varmistamisessa hyödynnetään QDataStream-luokkaa, jonka avulla käsittelyyn saadaan kokonaisia viestejä.

Kun kokonainen viesti on saapunut käsittelijälle, sen hyötykuorman tarkistussumma ja koko tarkistetaan vielä otsikon tietoja vasten. Tämän jälkeen hyötykuorma, joka on istunnon tietokanta, tallennetaan kiintolevylle tilapäistiedostoon. Sekä tämä istunnon tietokanta että pitkä aikavälin tietokanta avataan yhdistämisen ajaksi.

Tietokantojen yhdistämisessä muutosten kohde on pitkän aikavälin tietokanta. Pitkän aikavälin tietokannasta päivitetään ne lämpökartat, jotka löytyvät myös istunnon tietokannasta. Pitkän aikavälin tietokannassa olevaan keräysaikamuuttujaan lisätään istunnon keräysaika. Lämpökartat ja käyttäjäpolut ladataan molemmista tietokannoista muistissa oleviin oliorakenteisiin yhdistämistä varten. Varsinaiset yhdistämismenetelmät on kuvattu tarkemmin aiemmin tässä dokumentissa.

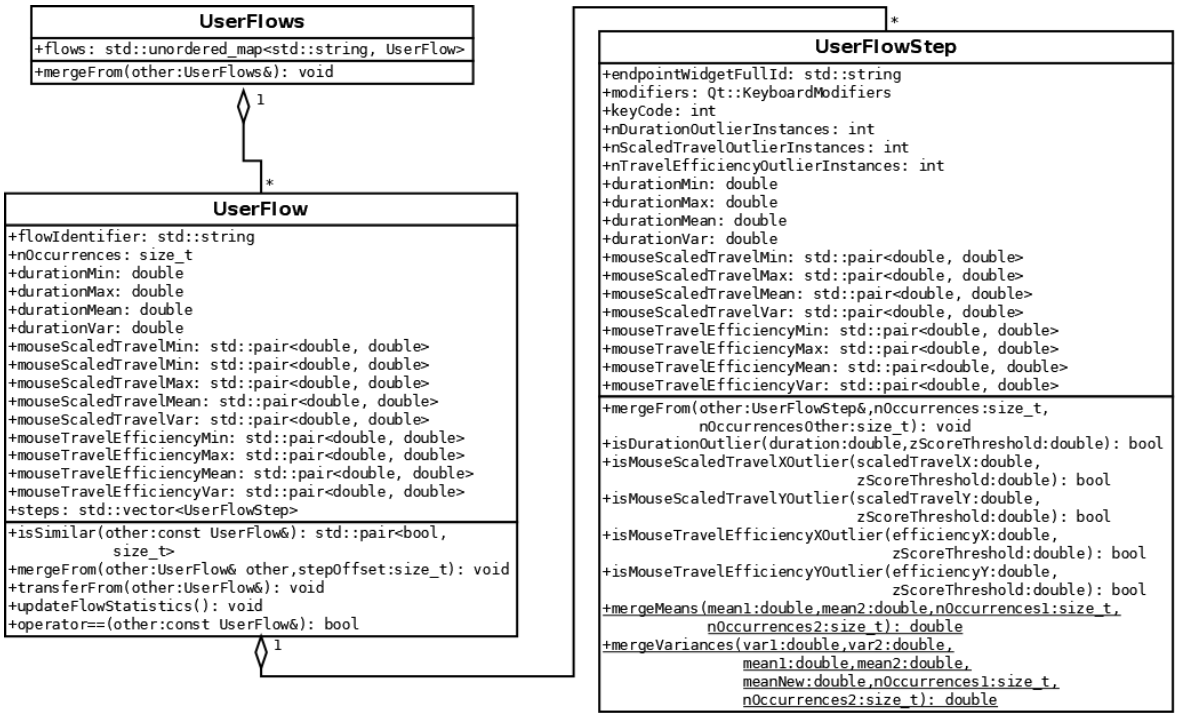
Analytiikkapalvelimen sovellus koostuu käytännössä vain yhdestä itse tehdystä luokasta, jonka Kuva 2 esittää karkeasti. Tästä näkee samalla mihin Qt-luokkiin tietokanta- ja tietoliikennetoiminnot perustuvat.



Kuva 2. Analytiikkapalvelinsovelluksen rakenne

7.6 Käyttäjäpolkujen tietorakenne

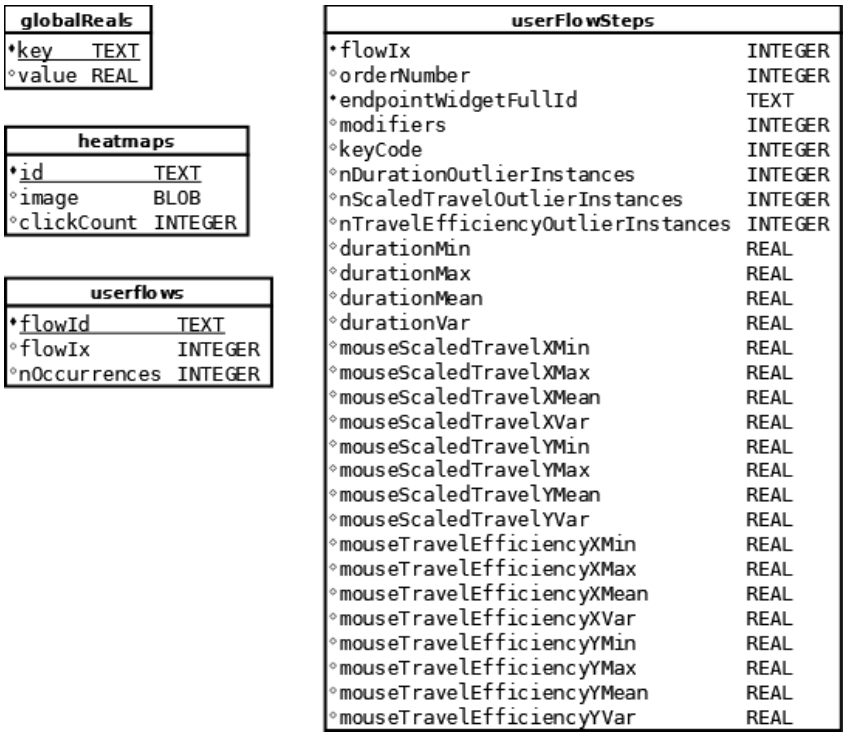
Käyttäjäpolkujen tietorakenne on analytiikkajärjestelmän keskiössä ja sitä käytetään sekä sovelluksen lisäosassa että palvelimessa. Tietorakenteen luokat ovat oleellisimmilta osin esitetty kuvassa 3. Luokkien tehtävä on käyttäjäpolkujen ja niiden askelten varastoinnin lisäksi tarjota toiminnot näiden samanlaisuuden tarkistamiseen ja yhdistämiseen. Lisäksi käyttäjäpolun tasolla on funktio, joka luo käyttäjäpolkutason tilastolliset tunnusluvut askeltason tilastollisista tunnusluvuista.



Kuva 3. Käyttäjäpolkujen tietorakenne

7.7 Tietokannat

Analytiikkajärjestelmässä käytetään yhdenlaista tietokantarakennetta useammassa tarkoituksessa. Käyttöliittymäsovelluksen analytiikkalisäosa käyttää paikallisesti kahta tietokantaa: istunnon tietokantaa ja pitkän aikavälin tietokantaa. Analytiikkapalvelin käyttää ylläpitämäänsä pitkän aikavälin tietokantaa ja tilapäisesti yhdistämistä varten tallentamaansa istunnon tietokantaa. Rakenne on kaikissa tietokannoissa sama ja se on esitetty kuvassa 4. Taulujen välille ei ole tarpeen asettaa relaatioita, koska ainut riippuvuus on käyttäjäpolusta sen askeleeseen ja haku tehdään aina halutun askeleen järjestysnumeron perusteella.



Kuva 4. Tietokantojen rakenne

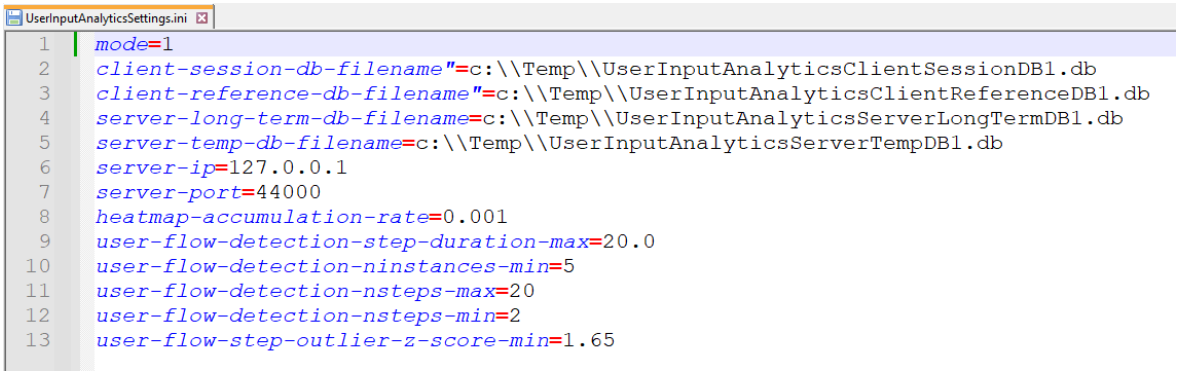
Tietokantamoottorina toimii SQLite, jolle Qt:ssa on helppokäyttöinen tuki (Katecpp 2015). SQLite, joka ei tarjoa palvelintoiminnallisuutta, soveltuu käyttöön myös, koska sitä ei ole tarve käyttää monen prosessin toimesta yhtä aikaa. Lisäksi SQLiten yhden tiedoston toteutus sopii hyvin tietokannan atomiseen siirtämiseen. Pelkkä SQL-palvelin ei myöskään riittäisi, koska analytiikkapalvelimessa on toiminnallisuus istuntojen yhdistämiseksi. Tietokannan siirrossa tarkistetaan aina tarkistussumma ja koko. Molemmat luvut ovat sanoman otsikossa ja niitä verrataan vastaanotossa binäärimuotoisesta tietokantatiedostosta laskettuihin lukuihin. Kaikkien tietokantojen rakenne on sama ja sisällöt ovat luettavissa SQL-kyselyillä perustietotyyppeinä pois lukien lämpökartat, jotka ovat PNG-muodossa. Tästä syystä dataan pääsee helposti käsiksi ulkopuolisestakin järjestelmästä.

Tietokantojen tyhjentäminen tai nollaaminen tehdään tällä hetkellä yksinkertaisesti poistamalla kyseinen tiedosto tiedostojärjestelmästä. Tietokantatiedostot voi poistaa myös sovellusten ollessa käynnissä, mutta tällöin on hyvä tiedostaa muutama seikka; Käyttöliittymäsovelluksen analytiikkalisäosa käyttää tiedostoa käynnistyksessä ja juuri ennen sovelluksen sammumista, joiden aikana tiedosto on varattuna sovellukselle. Palvelinsovellus voi käyttää tietokantatiedostojaan milloin tahansa ollessaan käynnissä, joten tiedostojen poistamisen yrittäminen sen ajon aikana ei ole suotavaa. Tiedoston kirjoittaminen ja lukeminen ovat kuitenkin atomisia operaatioita, eikä ole mahdollista poistaa tiedostoa kirjoituksen ollessa käynnissä, joten tiedoston korruptoitumisesta ei pitäisi olla vaaraa. Laitteen

sammuttaminen kesken kirjoittamisen puolestaan korruptoisi tietokannan lähes varmasti, mikä aiheuttaisi sen hylkäämisen seuraavalla kerralla, kun sovellusta käynnistetään.

7.8 Konfigurointi ja asetukset

Sovelluksen asetukset määritetään asetustiedostossa, jonka nimi annetaan sekä käyttöliittymäsovellukselle että palvelinsovellukselle käynnistysparametrissa user-input-analytics-settings-filename. Käynnistysparametri luetaan käyttämällä QCommandLineParser-luokkaa ja parametrien käsittely tapahtuu QSettings-luokan avulla. Kuva 5 esittää tiedostorakenteen, jota käytetään sekä käyttöliittymäsovelluksen että palvelinsovelluksen konfigurointiin.



Kuva 5. Asetustiedoston rakenne

Asetustiedostossa määritetään toimintatila, SQLite-tietokantojen tiedostosijainnit, analytiikkapalvelimen yhteystiedot, lämpökarttojen kerrytysnopeus ja parametrit käyttäjäpolkujen tunnistustoiminnolle.

8 Tulosten visualisointi ja demosovellus

8.1 Yleistä

Visualisoinnin ydin muodostuu vertailun avulla, jolloin johdonmukaisuus on tärkein yleisperiaate visualisoinnin suunnittelussa. Tietopisteiden mielekkään vertailun helpottamiseksi on välttämätöntä, että niiden esitystapa noudattaa yhtenäistä logiikkaa. Edward Tufte on muotoillut asian näin: ”Esitä muutos aineistossa, älä muutosta esitystavassa”. (Koponen ym. 2019, 42). Esitystapa tulisi olla linjassa koko julkaisun laajuudelta (Koponen ym. 2019, 43).

Tämä analytiikkajärjestelmä tukee tiedon vertailua monella tavalla sekä numeerisesti että visuaalisesti. Lämpökarttojen värit ja intensiteetit ovat vertailukelpoisia käyttöliittymäkomponenttien sisällä ja niiden välillä. Käyttäjäpolkujen ja niiden askelten tilastolliset tunnusluvut ovat vertailukelpoisia keskenään. Käyttöliittymäkomponenttien päälle tehtävät visualisoinnit lisätään läpinäkyvään ikkunaan epäsuorasti, koska käyttöliittymäkomponenttien piirtofunktiot ja piirtolaitteet (QPaintDevice) eivät ole käytettävissä erillisessä visualisointisäikeessä. Käyttöliittymäkomponenttien lämpökartat, askelnumerot ja klikkausten lukumäärät on mahdollista visualisoida vasta kun varjomalli on olemassa, joten tietokannan lataamisen jälkeen niitä pidetään odottamassa, kunnes kyseistä käyttöliittymäkomponenttia on käytetty ja sen varjomalli on luotu.

Näitä tuloksia esitellään tässä työssä kehitetyn ja seuraavissa kappaleissa kuvatun demosovelluksen avulla. Demosovelluksen tarkoitus on toimia todellisen tutkittavan käyttöliittymäsovelluksen asemessa ja mahdollistaa analytiikkajärjestelmän toimintojen esittely. Tämän tutkimuksen varsin suppea empiirinen osuus toteutettiin kehitetyllä demosovelluksella, joka on Qt-esimerkkisovelluksiin perustuva yksinkertainen, muutamia ikkunoita sisältävä, mutta kuitenkin täysin toimiva käyttöliittymäsovellus.

8.2 Lämpökartat

Tässä opinnäytetyössä on toteutettu käyttöliittymäkomponenttikohtaiset ja resoluutioriippumattomat lämpökartat hiiren osoittimen liikkeen ja hiiren klikkausten perusteella. Näitä lämpökarttoja on mahdollista tarkastella resoluutio- ja skaalariippumattomasti todellisen käyttöliittymän päällä yksitellen tai yhdessä. Lämpökartat käyttävät värigradientteja esittämään tietoarvojen vaihtelut spatiaalisesti. Hiiren osoittimen sijainti kuvataan punaisen värikomponentin voimakkuutena ja hiiren klikkaukset vihreän värikomponentin voimakkuutena.

Värien valinta ja käyttö lämpökartassa vaikuttaa oletettavasti siihen, miten käyttäjät tulkitsevat ja ymmärtävät esitettäviä tietoja. Lämpökarttojen väridynamiikka on tärkeä tiedon

välittämiseksi tehokkaasti ja oivallusten herättämiseksi datakuvioista. Värikomponenttien tietotyypin ollessa 32-bittinen liukuluku, dynamiikka on oletettavasti riittävän suuri.

Visualisointitilassa lämpökarttojen pikselien väriarvot normalisoidaan komponenteittain piirtämistä varten siten, että kunkin komponentin maksimi globaalisti on kaikkien käyttöliittymäkomponenttien osalta 1.0. Tällöin lämpökartan lyhyempikin keräysaika antaa näkyvän lopputuloksen, mutta tulos ei ole enää vertailukelpoinen istuntojen välillä. Jatkokehityskohteena olisikin hyvä, jos visualisointitilassa voisi normalisoinnin voisi aktivoida halutessaan. Huomionarvoista on, että normalisoidutkaan kuvat eivät lähde visualisointitilassa analytiikkapalvelimelle yhdistettäväksi pitkän aikavälin lämpökarttoihin.

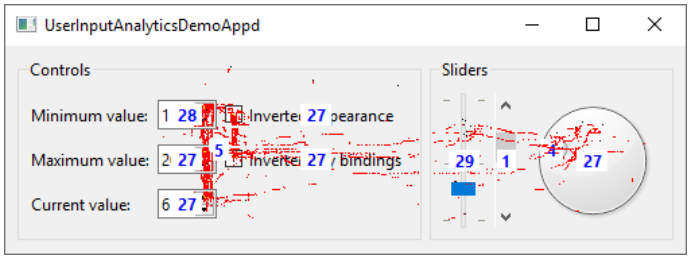
Resoluutioriippumattomuus on tärkeä tekijä datan yhdistämisessä, siirrettävyydessä käyttöliittymäympäristöstä toiseen ja vertailussa aikaisempien kuvien kanssa. Näitä tukemaan jokaiselle käyttöliittymäkomponentille määritetään oma XY-avaruutensa, jossa kyseisen komponentin lämpökarttaosuus säilytetään. Lämpökartan visualisoinnin laatua parantaa oletettavasti myös simultaanikontrastin huomioiminen. Simultaanikontrastin huomioiminen vaatisi kuitenkin visualisoinnissa jonkin verran optimoitua laskentaa ja sen tuoma hyöty korostuu vasta aktiivisemmassa käytössä, joten sen toteuttaminen on jätetty jatkokehityksen piiriin.

Lämpökarttoja visualisoitaessa niitä ei voi piirtää käyttöliittymäkomponenttikohtaisesti piirtotapahtumasignaalin (QPaintEvent) havaitsemisen jälkeen, jolloin piirron z-järjestys ja siten näkyvyys tulisi automaattisesti käsiteltyä. Tämä ei onnistu, koska kyseinen elementtikohtainen tapahtumasignaali saadaan juuri ennen kuin elementille kutsutaan piirtometodia (paintEvent(…)) ja näinollen lämpökartta piirtyisi yli elementin sovellusgrafiikalla. Normaalisti tehtävä piirtometodin ylikirjoittaminen (C++ overloading) ei ole tässä mahdollista, koska tämän visualisointitoiminnallisuuden on tarkoitus toimia niin sanotusti päälle liimattuna (add-on) minkä tahansa Qt-käyttöliittymäsovelluksen kanssa.

Lämpökartat ovat puoliläpinäkyviä ja ne visualisoidaan alpha-sekoittamalla ne yhteen, läpinäkyvään ja koko ruudun peittävään ikkunaan. Lämpökartan piirtäminen tuo merkittävän lisäkuorman, joten sitä päivitetään harvemmalla syklillä kuin tavallista näyttögrafiikkaa. Puoliläpinäkyvyys sekoittuu oikein, koska hiiritapahtumat taltioidaan vain päällimmäisen käyttöliittymäelementin lämpökarttaan. Tämä ei kuitenkaan päde visualisointitilassa pääikkunoiden välillä, koska ne ovat siirrettävissä toistensa päälle. Z-järjestyksen jättäminen huomiotta ei kuitenkaan haittaa, kunhan eri pääikkunoita ei tarkastella päällekkäin tuloksia katsottaessa.

Hiiren liikkeen ja klikkauspisteiden visualisoinnin lisäksi kehitetty analytiikkalisäosa mahdollistaa myös hiiren klikkauskertojen visualisoinnin. Havaittujen klikkausten lukumäärä

visualisoidaan haluttaessa numerona käyttöliittymäkomponentin keskellä. Alla on demosovellusta käyttämällä tuotettu esimerkki (Kuva 6) lämpökarttojen visualisoinnista.



Kuva 6. Lämpökarttojen visualisointi klikkausmäärillä

8.3 Käyttäjäpolut

Tunnistettujen käyttäjäpolkujen tiedot visualisoidaan taulukoilla, laatikko- ja viivakaavioilla ja käyttöliittymän päällä näytettävillä laskuritiedoilla. Visualisointikäyttöliittymän päänäkymä on taulukko, jossa näkyy sarakkeiden mukaan järjestettävä taulukko käyttäjäpoluista. Taulukon rivin valinnasta päivitetään toinen, käyttäjäpolun askelten tiedot sisältävää taulukkoa. Lisäksi käyttäjäpolun valinnan perusteella päivitetään erillisissä näkymissään olevia, valitun käyttäjäpolun askelten tilastollisia tunnuslukuja kuvaavia laatikko- ja viiksikaavioita.

Käyttäjäpoluilla ja niiden askelilla on toisistaan hieman eriävä taulukollinen esitystapa. Jokaiselle käyttäjäpolulle näytetään käyttäjäpolun tunniste (Id), tunnistettujen ilmentymien lukumäärä (Instances) sekä tilastolliset tunnusluvut käyttäjäpolun kestosta, hiiren osoittimen skaalatusta kulkumatkasta ja hiiren osoittimen kulkumatkan hyötysuhteesta.

Käyttäjäpolun askeleelle puolestaan visualisoidaan askelen päätepisteen käyttöliittymäkomponentin tunniste (Endpoint widget), keston poikkeamien lukumäärän (NTOIs), hiiren osoittimen skaalatun kulkumatkan poikkeamien lukumäärän (NSTOIs) ja hiiren osoittimen kulkumatkan tehokkuuden poikkeamien lukumäärän (NTEOIs). Lisäksi askelkohtaisesti näytetään vastaavat tilastolliset tunnusluvut kuin käyttäjäpolulle. Nämä tiedot päivittyvät visualisointikäyttöliittymässä käyttäjäpolkuvalinnan seurauksena.

Tilastollisia tunnuslukuja ovat minimi (*Min), maksimi (*Max), keskiarvo (*Mean), keskihajonta (*SD), ensimmäisen kvartiilin estimaatti (*Q1Est) ja kolmannen kvartiilin estimaatti (*Q3Est). On huomattava, että datan oletetaan olevan normaalijakautunutta, jolloin on mielekästä esittää *arviot* kvartiileista ja nämä lasketaan (kaavat 7-9, Taboga 2021) visualisointia varten seuraavasti:

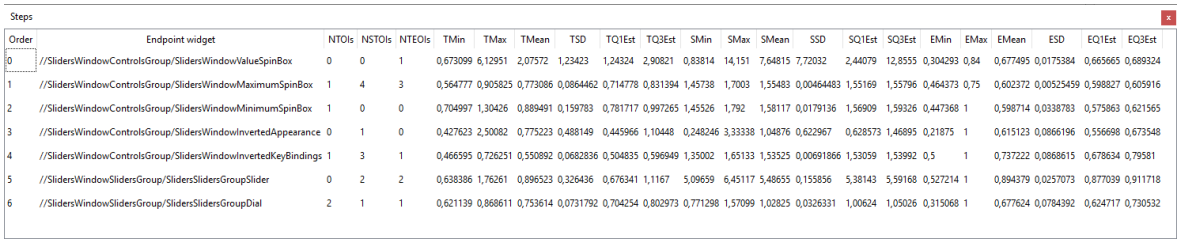
$$Q_1 = \mu - 0.6745 * \sigma \qquad (7)$$

$$Q_2 = \mu \quad (8)$$

$$Q_3 = \mu + 0.6745 * \sigma \quad (9)$$

Alla oleva kuva 7 esittää käyttäjäpolkujen listauksen, jossa on valittuna eniten ilmentymiä omaava käyttäjäpolku. Kuvassa näkyy myös laatikko- ja viiksikaaviota valitun käyttäjäpolun askelten metriikoista. 8 esittää saman käyttäjäpolun askelten tiedot taulukkomuodossa. Kuvista nähdään muun muassa, että käyttäjäpolun ensimmäinen askel on kestoltaan selvästi pisin ja sen kestossa on myös suurin vaihtelu. Tämä saattaisi selittyä esimerkiksi sillä, että käyttäjäpolun valinta vaatii käyttäjältä enemmän ajatustyötä kuin, mahdollisesti tutun, käyttäjäpolun suorittaminen sen jälkeen. Hiiren osoittimen kulkeman matkan hyötysuhde päinvastoin näyttää kasvavan käyttäjäpolun myöhemmissä askelissa, joka voisi johtua esimerkiksi keskittymisen herpaantumisesta. Poikkeamia on eniten (4 kappaletta) käyttäjäpolun toisessa askeleessa hiiren kursorin kulkeman skaalatun matkan arvoissa. Näiden lukujen ja kaavioiden tulkitseminen vakavasti edellyttäisi kuitenkin paljon suurempaa otantaa ja myös laadullisten menetelmien käyttämistä.

Kuva 7. Käyttäjäpolun visualisointi taulukkomuodossa ja viiksikaavioina, esimerkki 1

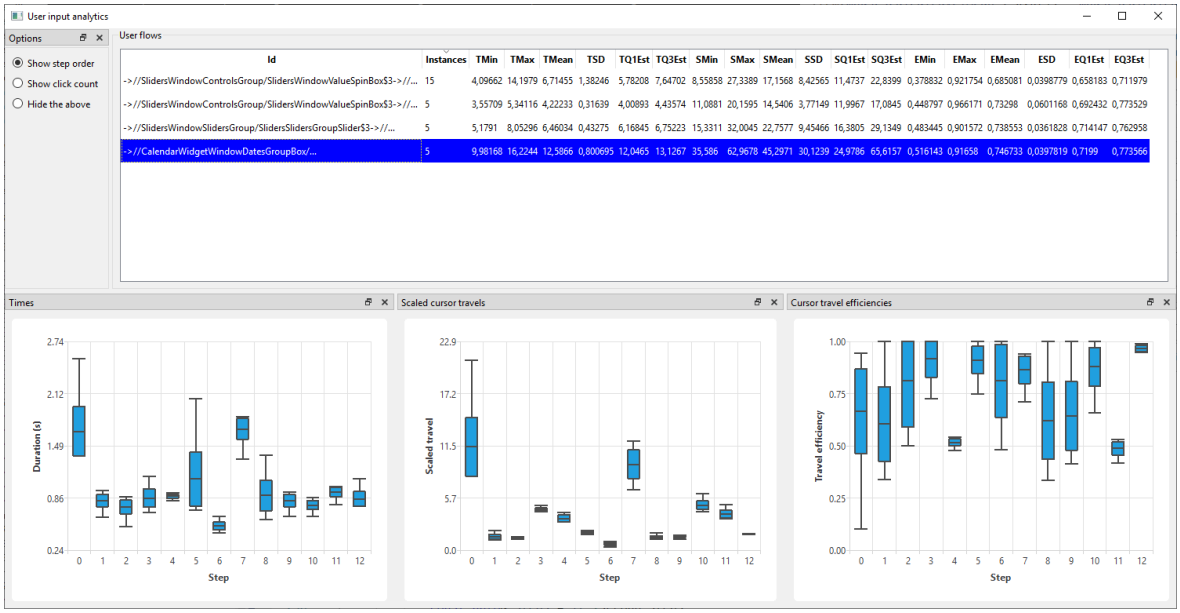


Steps

Order	Endpoint widget	NTOIs	NSTOIs	NTEOIs	TMin	TMax	TMean	TSD	TQ1Est	TQ3Est	SMin	SMax	SMean	SSD	SQ1Est	SQ3Est	EMin	EMax	EMean	ESD	EQ1Est	EQ3Est
0	//SlidersWindowControlsGroup/SlidersWindowValueSpinBox	0	0	1	0,673099	6,12951	2,07572	1,23423	1,24324	2,90821	0,83814	14,151	7,64815	7,72032	2,44079	12,8555	0,304293	0,84	0,677495	0,0175384	0,665665	0,689324
1	//SlidersWindowControlsGroup/SlidersWindowMaximumSpinBox	1	4	3	0,564777	0,905825	0,773086	0,0864462	0,714778	0,831394	1,45738	1,7003	1,55483	0,00464483	1,55169	1,55796	0,464373	0,75	0,602372	0,00525459	0,598827	0,605916
2	//SlidersWindowControlsGroup/SlidersWindowMinimumSpinBox	1	0	0	0,704997	1,30426	0,889491	0,159783	0,781717	0,997265	1,45526	1,792	1,58117	0,0179136	1,56909	1,59326	0,447368	1	0,598714	0,0338783	0,575863	0,621565
3	//SlidersWindowControlsGroup/SlidersWindowInvertedAppearance	0	1	0	0,427623	2,50082	0,775223	0,488149	0,445966	1,10448	0,248246	3,33338	1,04876	0,622967	0,628573	1,46895	0,21875	1	0,615123	0,0866196	0,556698	0,673548
4	//SlidersWindowControlsGroup/SlidersWindowInvertedKeyBindings	1	3	1	0,466595	0,726251	0,550892	0,0682836	0,504835	0,596949	1,35002	1,65133	1,53525	0,00691866	1,53059	1,53992	0,5	1	0,737222	0,0868615	0,678634	0,79581
5	//SlidersWindowSlidersGroup/SlidersSlidersGroupSlider	0	2	2	0,638386	1,76261	0,896523	0,326436	0,676341	1,1167	5,09659	6,45117	5,48655	0,155856	5,38143	5,59168	0,527214	1	0,894379	0,0257073	0,877039	0,911718
6	//SlidersWindowSlidersGroup/SlidersSlidersGroupDial	2	1	1	0,621139	0,868611	0,753614	0,0731792	0,704254	0,802973	0,771298	1,57099	1,02825	0,0326331	1,00624	1,05026	0,315068	1	0,677624	0,0784392	0,624717	0,730532

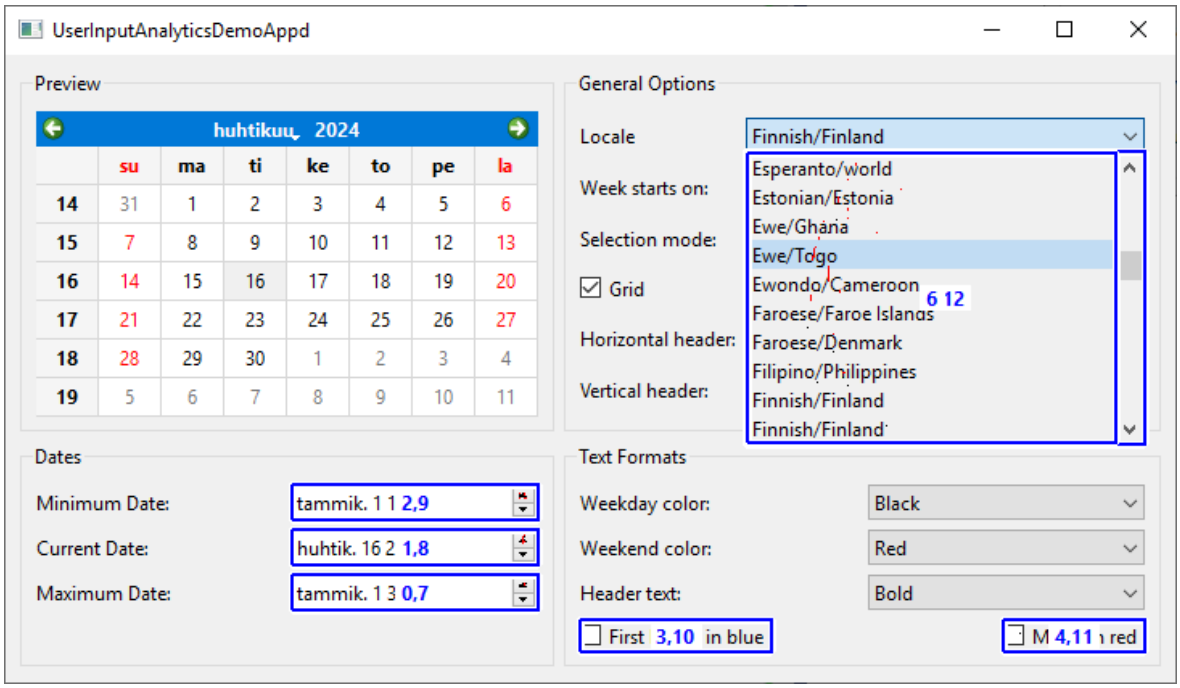
Kuva 8. Käyttäjäpolun askelten visualisointi taulukkomuodossa, esimerkki 1

Kuvassa 9 esitetään suppeana esimerkkinä toisen käyttäjäpolun valinnan seurauksena nähtävät laatikko- ja viiksikaaviot. Tässä näkyy sama tilanne kuin alempana (kuvassa 10), jossa käyttäjäpolkuun on tunnistettu samojen käyttöliittymäkomponenttien toistuvaa käyttöä. Käyttäjäpolku näyttää sisältävän kahdesti toistuvan askelsekvenssin, joka saattaisi laadullisessa tarkastelussa osoittautua kahdeksi erilliseksi käyttäjäpoluksi, joista ensimmäinen sisältäisi yhden askeleen toista enemmän. Tämän esimerkin otannan koko on vain viisi.



Kuva 9. Käyttäjäpolun visualisointi taulukkomuodossa ja viiksikaavioina, esimerkki 2

Käyttäjäpolun askelten järjestysnumerot ovat visualisoitu käyttöliittymäkomponentin keskelle, kuten alla olevasta demosovelluksen käytön esimerkkikuvasta 10 nähdään. Kuvasta nähdään myös, että käyttäjäpolussa samaa käyttöliittymäkomponenttia käytettäessä useammin kuin kerran, eri käyttökertojen lukemat erottuvat pilkulla. Tämän visualisoinnin voi valita näkymään valintaikkunassa edellä esitetyissä kuvissa 7 ja 9 näkyvällä valinnalla (Options-ikkunassa).



Kuva 10. Käyttäjäpolun askelten järjestysnumeroiden visualisointi

Hiiren osoittimen kulkemien skaalattujen matkojen tilastolliset tunnusluvut tallennetaan ja käsitellään tässä analytiikkajärjestelmässä x- ja y-komponentti erillään, mutta visualisoinnin selkeyttämiseksi ne yhdistetään yhdeksi luvuksi Pythagoraan lausetta käyttämällä. Myös hiiren osoittimen kulkemien matkojen hyötysuhteiden tilastolliset tunnusluvut tallennetaan ja käsitellään vastaavasti, mutta visualisointia varten ne yhdistetään Pythagoraan lauseen sijaan laskemalla niistä keskiarvo.

9 Yhteenveto ja pohdinta

9.1 Yleistä

Opinnäytetyössä pohdittiin ensiksi sovelluksen käyttökokemuksen merkitystä pitäen silmällä toimeksiantajan tarpeita. Käyttökokemuksella tunnistettiin olevan erityistä merkitystä markkinoinnissa, loppukäyttäjän arjessa ja tuotekehittäjien työlle. Markkinoinnin alueella satunnaisten tuotteen käyttäjien saama käyttökokemus todettiin merkitykselliseksi sekä tuotteen maineen että yrityksen brändin kannalta. Loppukäyttäjälle käyttökokemuksen nähtiin tarkoittavan käytön helppoutta, virheettömyyttä ja tuottavuutta, jotka lisäävät pitkäaikaista sitoutumista ja jaksamista työssä. Tuotteen kehittäjille monimutkaisemman tuotteen käyttökokemuksen pääteltiin vaikuttavan vahvasti tuotteen kehittämiseen, koska hyvin suunnitellun käytettävyyden ajateltiin mahdollistavan osuvammat palautteet ja korjausliikkeet jo tuotekehityksen aikana.

Jo työtä aloitettaessa tiedettiin käyttökokemuksen parantamiselle olevan tarvetta ja samalla myös tunnistettiin määrällisten menetelmien joukossa olevan hyödyntämätöntä potentiaalia. Tuotteen käyttökokemuksen parantamisen todettiin mahdollisesti hyötyvän määrällisten tutkimusmenetelmien käytöstä laadullisen käyttötutkimuksen tukena. Määrällisen käyttötutkimuksen tuloksia ehdotettiin hyödynnettäväksi esimerkiksi toteutettaessa kognitiivisen läpikäynnin menetelmää, jolloin tulokset auttaisivat suuntaamaan huomion oikeisiin kohtiin käyttöliittymässä ja käyttötavoissa. Tämän ohessa pohdittiin myös laajemmin käyttöanalytiikan roolia osana tuotekehitysprosessia.

Käyttöliittymän- ja kokemuksen määrällistä tutkimusta on tehty kauan ja laajasti maailmalla. Empiirinen katsaus tärkeimpien käyttöanalytiikkaohjelmistojen markkinatilanteeseen ja toiminnallisuuksiin osoitti kuitenkin aukon sekä soveltuvien analytiikkatyökalujen, että analyysimenetelmien valikoimissa. Suurimman osan työkaluista todettiin myös toimivan muissa kuin työpöytäympäristössä. Käyttökelpoisimmaksi tuotteeksi osoittautui toimeksiantajan ohjelmistoympäristöä tukeva analytiikkajärjestelmä Qt Insight, josta kuitenkin löytyi ratkaisevia rajoitteita ja kehitystarpeita. Qt Insight -tuotteen soveltumista rajoitti erityisesti pilvipalvelupohjaisuus, keskeneräinen analytiikka ja sen tarjoaman tietosuojan otaksuttu vähyys.

Näihin lähtökohtiin perustuen nähtiin tarpeellisuus ja mielekkyys opinnäytetyön varsinaiselle aiheelle, jossa kehitettiin erillisverkossa toimiva käyttödatan keruu-, analysointi- ja visualisointijärjestelmä Qt-pohjaisten käyttöliittymäsovellusten käytön ja käytettävyyden tutkimiseen. Opinnäytetyössä onnistuttiin kehittämään yksityisessä verkossa ilman pilvipalvelua toimiva käyttödatan keruu-, analysointi- ja visualisointialusta Qt-pohjaisten

käyttöliittymäsovellusten käytön ja käytettävyyden tutkimiseen. Järjestelmään myös kehitettiin ja toteutettiin muutamia tarpeellisimmiksi arvioituja analytiikkamenetelmiä.

Tapahtumapohjaisen käyttöliittymän analytiikkajärjestelmän voi todistetusti tehdä sellaiseksi, ettei tapahtumia tarvitse viedä pois käyttäjän työasemalta, mutta järjestelmä voi kuitenkin oppia keskitetysti (engl. federated learning). Tilastollisia tuloksia tallentavana ja siirtävänä järjestelmänä tämä selvästi paitsi parantaa käyttäjän tietosuojaa, myös mahdollistaa järjestelmän skaalautumisen käytännössä rajattoman suurelle käyttäjäjoukolle ja keräysajalle. Analytiikka on helppo lisätä olemassa olevan käyttöliittymäsovelluksen suoritukseen toteutetulla plugin-ratkaisulla. Karkeasti ottaen tässä opinnäytetyössä uutta on erilainen tapa varastoida ja käsitellä tapahtumia, käyttäjäpolkujen automaattinen tunnistaminen visualisointeineen ja dynaamiset resoluutioriippumattomat lämpökartat. Myös käyttöliittymän rakenteen rekonstruointi analytiikassa ja tulosten näyttäminen analysoitavan sovelluksen päällä ovat edistyksellisiä toimintoja. Tärkeänä huomiona toteutuksessa nousi esiin se, ettei kertaalleen analyysiin käytettyä dataa käytetä uudelleen muuten kuin poikkeamien (engl. outliers) tunnistamiseen käyttäjäpoluissa.

Kehitetty analytiikkajärjestelmä täydentää Qt-pohjaisten sovellusten käyttöliittymän ja -kokemuksen tutkimusmenetelmiä internetistä ja pilvipalveluista riippumattomaan suuntaan. Järjestelmä tarjoaa lisäksi pohjan paitsi laajemmin tekoälyä soveltavalle analytiikkatoiminnallisuudelle, myös älykkäille käyttäjän avustamis- ja opastustoiminnoille. Järjestelmän odotetaan ensivaiheessa helpottavan muun muassa yhteisten syiden löytämistä ongelmatilanteille ja käyttötiedon kulkeutumista systemaattisemmin kehitystiimille asti. Kokonaisuudessaan tämä koettiin uudeksi ja mielenkiintoiseksi avaukseksi Patrialla.

Opinnäytetyö tehtiin hyvin itsenäisesti ja kehitettyyn analytiikkajärjestelmään ei tullut sidoksia Patriaan, sen tuotteisiin, toimintaan tai asiakkaisiin. Analytiikkajärjestelmää ei vielä aikataulusyistä käytetty Patrialla tuotteen ja käytön tutkimiseen pientä kokeilua lukuun ottamatta, vaan tyydyttiin osoittamaan kehitetyn järjestelmän toiminta demosovelluksen avulla. Työllä luotiin edellytykset käyttöanalytiikalle, mutta käyttöönotto Patrialla vaatii käyttöönottopäätöksen lisäksi teknisiä valmisteluja, kuten käyttöliittymäkomponenttien kattavan nimeämisen. Kynnyskysymyksenä on lisäksi asiakkaiden vakuuttuminen esimerkiksi analytiikan tarpeellisuudesta ja turvallisuudesta. Patrialla työ on nähty uutena avauksena ja se on herättänyt mielenkiintoa. Patrian edustajan mukaan ”enemmän tuloksia ja analyysiä veisi työn seuraavalle tasolle”, mutta lyhyen keskustelun jälkeen hän yhtyi näkemykseen, että todellisen tuotteen käyttäminen demosovelluksena kasvaisi työmäärältään liian suureksi (Mattila 2024).

Qt Insight -tuotekehitystiimi oli valmis tukemaan analytiikan kehittämisessä ja avaamaan tuotteen sisäisiä rajapintoja opinnäytetyön tekemisen ajaksi. Toteutetun analytiikkajärjestelmän rakenne ja toimintaperiaate eroavat perustavanlaatuisesti Qt Insight -tuotteen vastaavista aina datan keräämisestä tulosten visualisointiin, jolloin heidän tukeaan tai sisäisiä rajapintojaan ei ollut tarpeen käyttää. Näiden rajapintojen käyttö olisi myös sitonut opinnäytetyön toteutuksen Qt:n yksityisomistukselliseen (IP) materiaaliin. Qt Companystä on vahvistettu, että työ on hyvin tehty ja mielenkiintoinen kokonaisuus, eikä pyörää ole keksitty uudestaan (Haantie 2024).

9.2 Mahdollisia jatkotutkimus- ja kehitysaiheita

Analytiikkajärjestelmän toteuttamisen lomassa nousi esille monenlaisia selvitys- ja parannusideoita, joita ei kuitenkaan ollut mahdollista toteuttaa työn aikataulun puitteissa. Tässä kappaleessa käsitellään jatkotutkimus- ja kehitysaiheita teemoittain: reflektoidaan opinnäytetyötä kunkin teeman näkökulmasta ja käsitellään lyhyesti näkymät jatkotutkimukselle. Opinnäytetyöstä käytännössä kokonaan erilliset jatkotutkimusaiheet listataan lopuksi lyhyesti.Tuki A/B-testaukselle

Yksi tällainen parannus olisi tuki A/B-testaukselle eri järjestelmien ja eri ohjelmistoversioiden kesken. Tämä tarkoittaisi lähinnä käyttöliittymäsovelluksen versiotiedon viemistä tietokantaan ja kykyä vaihtaa tietokanta nopeasti toiseksi analytiikan ohjauskäyttöliittymässä. Tällöin voisi myös tutkia lämpökarttojen samanlaisuutta numeerisen arvon (korrelaatio) avulla. Käyttäjäpolkujen askelten laatikko- ja viiksikaavioita voisi myös näyttää rinnakkain tai animoida vaihdettaessa.

Käyttäjäpalautteen kerääminen

Käyttäjäpalautteen kerääminen analytiikkalisäosaa käyttäen saattaisi olla hyödyllinen toiminto, jolla käyttäjä voisi antaa kontekstisidonnaista palautetta heti tarpeen vaatiessa. Palautteenantomekanismia ei tarvitsi luoda sovellukseen erikseen ja palautteen voisi antaa esimerkiksi yleisessä muodossa käyttökokemuksen pisteytyksenä tai tekstimuotoisena. Annettua palautetta olisi mahdollista tarkastella analyysissä myös mitatun numeerisen tiedon valossa, jolloin sen konteksti mahdollisesti täsmentyisi ja syiden löytäminen helpottuisi.

Parannukset visualisointiin

Esitystapaa käyttäjäpolkujen kokonaisajoista (Kuvio 1, sivulla 17) ei toteutettu, koska käyttäjäpolkujen määrä on helposti niin suuri, että taulukko osoittautui käyttökelpoisemmaksi. Sen sijaan esimerkiksi viisi pisintä voisi näyttää. Tulosten visualisointia voisi mahdollisesti parantaa myös soveltamalla monimuuttujatiedon kuvaamista ja toistoperiaatetta (Koponen

ym. 2019, 58). Käyttäjäpolut voisi esittää toistokuvina tai kuvamatriisina (Koponen ym. 2019, 59). Myös erilaisia kuvaamisen rakennetyyppejä (Koponen ym. 2019, 62-64) voisi pyrkiä hyödyntämään tulosten visualisoinnissa.

Visualisoinnin suorituskyvyssä olisi optimoitavaa esimerkiksi piirtotapahtumien hyödyntämisen osalta. Tapahtumakäsittelijä voisi merkitä varjomalleille piirtotarpeen piirtotapahtumien perusteella, jolloin läpinäkyvässä ikkunassa piirrettäisiin vain päivittyneiden komponenttien grafiikat. Tällä hetkellä piirtotapahtumia ei hyödynnetä, vaan kaikkien näkyvissä olevien komponenttien visualisointi tehdään ajastettuna sekunnin välein.

Keräystilassa lämpökarttojen korkeimman resoluution säilyttäminen taustalla parantaisi tiedon säilyvyyttä. Nyt lämpökartat interpoloidaan aina käytetylle resoluutiolle ja skaalattaessa niitä pienemmäksi informaatiota häviää. Voisi myös olla hyödyllistä tarjota mahdollisuus deaktivoida lämpökarttojen normalisointi ennen visualisointia. Samalla tässä voisi toteuttaa kuvien väritasojen kvantisoinnin simultaanikontrasti-ilmiön hyödyntämiseksi. Simultaanikontrasti tulisi huomioida analytiikkalisäosassa kvantisoimalla lämpökartan väriarvot seitsemään tasoon pikselikohtaisesti juuri ennen näyttämistä.

Simultaanikontrasti on visuaalinen ilmiö, joka näkyy, kun ympäröivät värit vaikuttavat yhden värin havaitsemiseen. Tämä ilmiö johtaa kohteen havaitun värin ylikorostumiseen tai muuttumiseen, kun sitä tarkastellaan muiden värien rinnalla. Värien vaikutus on selkein, kun ympäröivät värit ovat täydentäviä tai kontrastisia kohteen värin kanssa. Ihminen kykenee tunnistamaan luotettavasti useimmiten vain kuusi tai seitsemän saman värin tummuusastetta. (Koponen ym. 2019, 49.)

Käyttöliittymän rakenteen ja monimutkaisuuden selvittäminen

Sovelluksen käyttöliittymän rakennetta voi selvittää analysoimalla sovelluksen käyttöliittymää ja tutkimalla sen eri osa-alueita. Näkymiä, painikkeita, valikkoja ja muita käyttöliittymäkomponentteja voi tarkastella sekä selvittää, miten ne on organisoitu ja miten ne toimivat yhdessä. Tämän voi tehdä myös automaattisesti seuraamalla käyttöliittymäympäristön tapahtumia, jos ne ovat saatavilla. (Riegler ja Holzmann 2018, 210-211.)

Käytännössä ohjelman suorituksen aikana tapahtumien perusteella selviää esimerkiksi luotavan käyttöliittymäkomponentin vanhempi (engl. parent), nimi, koko ja sijainti näytöllä. Näin saatava hierarkkinen puurakenne koko sovelluksesta on pohjana muulle käyttöanalytiikalle ja sitä voi hyödyntää myös karttana testauksen suunnittelussa ja muutostenhallinnassa.

Käyttöliittymän täyden rakenteen automaattinen selvittäminen ja kompleksisuuden mittaaminen rajattiin pois tämän opinnäytetyön piiristä työtä tehdessä sen oletetun laajuuden vuoksi, mutta se voisi olla mielenkiintoinen asia selvittää tulevaisuudessa. Nämä

toiminnallisuudet vaatisivat toimiakseen käyttöliittymän kaikkien komponenttien läpikäynnin ohjelman suorituksen aikana, koska näin saatu varjomallihierarkia kattaa vain sen osuuden käyttöliittymästä, jota on käytetty.

Käyttöliittymän rakenteellisen monimutkaisuuden mittaaminen voi sisältää sijoittelun, koostumuksen ja eri käyttöliittymäelementtien välisten suhteiden arvioimista. Mitattavia asioita voivat olla esimerkiksi käyttöliittymäkomponenttien lukumäärä, pienuus ja kohdistusvirhe (Riegler ja Holzmann 2018, 213-214). Lisäksi monimutkaisuuteen voisi olettaa vaikuttavana muun muassa hierarkiatasojen lukumäärä ja keskimääräinen käyttäjäpolkujen pituus. Kompleksisuuden kehittymistä ohjelmaversiosta toiseen voi olla hyödyllistä seurata (Riegler ja Holzmann 2018, 217), varsinkin jos absoluuttista monimutkaisuuden tasoa on vaikea mitata.

Yksinkertaisen ja intuitiivisen käyttöliittymän voi ajatella olevan käyttäjille helpompi ymmärtää ja navigoida. Sen voi myös olettaa loiventavan oppimiskäyrää tuovan ohjelman laajemman yleisön saataville. Monimutkainen käyttöliittymä puolestaan saattaa edellyttää käyttäjiltä enemmän aikaa sovelluksen opetteluun ja tutkimiseen. Tämä voi olla este uusille tai yksinkertaisista käyttöliittymistä pitäville käyttäjille.

Virtaviivainen ja suoraviivainen käyttöliittymä voi osaltaan lisätä käyttäjien tehokkuutta, koska käyttäjät voivat suorittaa tehtäviä nopeammin ja vähemmillä vaiheilla. Monimutkainen käyttöliittymä sen sijaan saattaa tarjota lisäominaisuuksia, mutta käyttäjien on ehkä navigoitava useiden tasojen tai vaihtoehtojen välillä, mikä saattaa vaikuttaa tuottavuuteen. Yleisen käsityksen mukaan käyttäjät arvostavat yksinkertaisuutta, ja suoraviivainen käyttöliittymä voi siten edistää positiivista käyttökokemusta ja tyytyväisyyttä. Jos käyttäjät pitävät monimutkaisuutta ylivoimaisena tai hämmentävänä, se voi johtaa turhautumiseen ja tyytymättömyyteen.

Yksinkertaiset käyttöliittymämallit ovat yleensä helpommin jalkautettavissa laajemmalle käyttäjäkunnalle, mukaan lukien eritasoisen teknisen asiantuntemuksen tai rajoitteen omaavat käyttäjät. Käyttäjät saattavat tehdä vähemmän virheitä yksinkertaisessa käyttöliittymässä, koska vuorovaikutuksessa ja ymmärtämisessä on vähemmän elementtejä. Monimutkaisempi käyttöliittymä voi lisätä käyttäjän virheiden todennäköisyyttä, varsinkin jos käyttöliittymäkomponenttien asettelu ei ole intuitiivinen tai jos käyttäjien on vaikea löytää joitakin ominaisuuksia.

Lopulta ihanteellinen monimutkaisuusaste riippuu kohdeyleisöstä, sovelluksen luonteesta ja käyttöliittymäsuunnittelun erityisistä tavoitteista. Toimivuuden ja yksinkertaisuuden tasapainottaminen on avainasemassa, kun tarjotaan positiivista käyttökokemusta. Ohjenuorana voisi pitää pyrkimistä yksinkertaisuuteen samalla kun täyttää toiminnalliset vaatimukset.

”Paras käyttöliittymä on sellainen, mitä ei huomaa”, toteaa Harri Heikkilä (Heikkilä ym. 2022, 45).

Käyttäjäryhmien ja -tyyppien tunnistaminen automaattisesti

Sovelluksen eri käyttäjäryhmien ja käyttötyylien tunnistaminen automaattisesti olisi mielenkiintoista, koska liittyvä tietomäärä on potentiaalisesti suuri ja tulokset hyödyllisempiä erityisesti silloin, kun käyttäjäpolut on huomioitu kattavasti. Eri käyttäjäryhmien ymmärtäminen mahdollistaa käyttöliittymän räätälöinnin kunkin ryhmän erityistarpeiden ja mieltymysten mukaan. Käyttöliittymän, ominaisuuksien ja sisällön mukauttaminen eri käyttäjäsegmenteille voisi odottaa lisäävän käytettävyyttä ja käyttäjien tyytyväisyyttä. Automaattinen ja reaaliaikainen tunnistus voisi myös toimia pohjana dynaamiselle käyttöliittymän mukauttamiselle ja kontekstuaalisen tiedon tarjoamiselle.

Käyttäjäryhmien ja -tyyppien tunnistaminen automaattisesti olisi luonteva sovelluskohde koneoppimiselle. Koneoppimisalgoritmeja voisi soveltaa käyttäjien käyttäytymismallien analysoinnissa ja ennusteiden tekemisessä käyttäjäryhmittäin tai -tyypeittäin. Tunnistettujen käyttäjäpolkujen joukkoa ja esimerkiksi ryhmittely- eli klusterointimenetelmiä voisi hyödyntää sovelluksen käyttäjätyyppien tai -ryhmien erottamisessa.

Tietoja tulisi kerätä anonyymisti (esimerkiksi käyttäjä ”A”, ”B”, ”C”...), eli käyttäjää ei yksilöitäisi, eikä esimerkiksi käyttäjätunnusta käyttäisi analyysissä. Tällä varmistettaisiin käyttäjien yksityisyyden kunnioittaminen ja asiaankuuluvien tietosuojamääräysten noudattaminen. Avoimuus olisi kuitenkin varmistettava kerättävien tietojen osalta ja käyttäjän suostumus olisi hankittava tarvittaessa. Huomionarvoista olisi myös se, että käytettävillä menetelmillä ei olisi tarkoitus seurata yksittäisen henkilön tuottavuutta, vaan tarkastella tuotteen soveltuvuutta erilaisille käyttäjä- ja tehtävätyypeille. Oleellinen kysymys tunnistettujen käyttäjäpolkujen hyödyntämisessä on se, missä kohdissa suunniteltu polku ja käyttäjän toiminta eroavat toisistaan (Heikkilä ym. 2022, 52).

Erilliset jatkotutkimusaiheet

Potentiaalinen jatkotutkimusaihe olisi selvittää millaisen parannuksen tutkimukseen toisi katseenseurannan (engl. eye tracking) käyttäminen käyttäjän syötteen analyysiin lisänä. Yksi uusi hyöty saattaisi olla niin sanotun roskagrafiikan tehokkaampi tunnistaminen. Käyttäjä saattaa silmäillä eri grafiikkaelementtejä paljonkin ilman että käy niissä hiiren osoittimella vierailemassa. Roskagrafiikka on Koposen ym. (2019, 78) mukaan esimerkiksi harhaanjohtavaa graafien koristelua, grafiikkaelementtien liioittelevaa skaalausta ja epäsäännöllisen muotoisia elementtejä.

Toinen mahdollisesti hyödyllinen jatkotyöaihe voisi olla käyttäjän hienovaraisen avustamisen tutkiminen. Käyttäjän hienovaraisessa avustamisessa voisi esimerkiksi väläyttää käyttäjäpolun todennäköisimmän seuraavan askeleen kontrollia tai sen kohdalla voisi muuttaa osoittimen ulkoasua tai vilauttaa työkaluvinkkiä. Tämä tarkoittaisi analytiikkatoiminnallisuuden integroimista varsinaiseen sovellukseen tai sovelluslisäosan kehittämistä siten, että se pystyisi lisäämään toivottuja efektejä sovelluksen käyttöliittymän päälle.

Kolmas mielenkiintoinen aihe voisi olla selvittää voisiko lämpökarttoja ja käyttäjäpolkuja visualisoida reaaliaikaisina ja soveltuisivatko ne silloin käytettäväksi esimerkiksi ryhmä- tai etäopetuksen tukena. Voisiko luennoija esimerkiksi tarkistaa nopeasti millaisia käyttäjäpolkuja oppilaat suorittavat ja huomioida ne heti opetuksessa.

Lähteet

Adobe. 2022a. UI Design. Viitattu 11.2.2024. Saatavissa https://xd.adobe.com/ideas/process/ui-design/

Adobe. 2022b. User flow diagram — what it is, why it's important, and how to create one. Viitattu 14.2.2024. Saatavissa https://business.adobe.com/blog/basics/how-to-make-a-user-flow-diagram

Balkhi, S. 2023. The Importance of User Feedback and Data in UX Design. Viitattu 11.2.2024. Saatavissa https://www.uxmatters.com/mt/archives/2023/12/the-importance-of-user-feedback-and-data-in-ux-design.php

Baskanderi, N. 2017. UX Glossary: Task Flows, User Flows, Flowcharts and some new-ish stuff. UX Planet. Viitattu 14.2.2024. Saatavissa https://uxplanet.org/ux-glossary-task-flows-user-flows-flowcharts-and-some-new-ish-stuff-2321044d837d

BBSysDyn. 2022. How do I combine standard deviations of two groups? Viitattu 16.2.2024. Saatavissa https://math.stackexchange.com/questions/2971315/how-do-i-combine-standard-deviations-of-two-groups

Berg, J. 2023. Branding and UX/UI Design: why is it important? Viitattu 14.2.2024. Saatavissa https://www.ironhack.com/gb/blog/branding-and-ux-ui-design-why-is-it-important

Browne, C. 2024. What are User Flows in User Experience (UX) Design? Careerfoundry. Viitattu 14.2.2024. Saatavissa https://careerfoundry.com/en/blog/ux-design/what-are-user-flows/

DataHeroes. 2024. Top 5 Outlier Detection Methods Every Data Enthusiast Must Know. Viitattu 19.4.2024. Saatavissa https://dataheroes.ai/blog/outlier-detection-methods-every-data-enthusiast-must-know/

DesignGuru. 2023. How to use Heat maps in UX design. Bootcamp. Viitattu 18.2.2024. Saatavissa https://bootcamp.uxdesign.cc/how-to-use-heat-maps-in-ux-design-6d264dbc61c7

Elements of AI. 2024. Koneoppimisen lajit. Viitattu 11.2.2024. Saatavissa https://course.elementsofai.com/fi/4/1

Eskola, O. 2008. Kognitiivinen läpikäynti. Viitattu 20.4.2024. Saatavissa https://oivaeskola.fi/2008/02/15/kognitiivinen-lapikaynti/

Finlex. 2004. Laki yksityisyyden suojasta työelämässä. Viitattu 25.4.2024. Saatavissa https://www.finlex.fi/fi/laki/ajantasa/2004/20040759

Haantie, T. 2023. RE: Tutkimusaihe. Sähköpostiviesti. Vastaanottaja Nikulainen, J. Lähetetty 15.6.2023.

Haantie, T. 2024. RE: Opinnäytetyö lukaistavaksi ja kommentoitavaksi. Sähköpostiviesti. Vastaanottaja Nikulainen, J. Lähetetty 7.5.2024.

Heikkilä, H., Markkanen P., Mustaniemi T., Pakarinen K., Piironen J., Torkkeli L. & Väisänen J. 2022. UX Opus. Opas käyttökokemuksen termiviidakkoon. LAB-ammattikorkeakoulun julkaisusarja, osa 55.

Hotjar. 2023. Using heatmaps to improve your website's UX. Viitattu 18.2.2024. Saatavissa https://www.hotjar.com/blog/improve-website-ux-with-heatmaps/

Katecpp. 2015. Sqlite with Qt – step by step. Viitattu 24.4.2024. Saatavissa https://katecpp.github.io/sqlite-with-qt/

Koponen, J., Hildén J., ja Vapaasalo T. 2019. Tieto näkyväksi. 3. painos. Keuruu: Otavan Kirjapaino Oy.

Lukkarinen, A. 2022, Teaching Event-driven Programming for Novice Programmers: Challenges and Advances. Viitattu 24.4.2024. Saatavissa https://aaltodoc.aalto.fi/server/api/core/bitstreams/f0238ab9-0e63-46d5-bfa4-ae38e3b864be/content

Macleod, S. 2023. Qualitative vs Quantitative Research Methods & Data Analysis. Viitattu 11.2.2024. Saatavissa https://www.simplypsychology.org/qualitative-quantitative.html

Mattila, K. 2024. RE: Opinnäytetyö lukaistavaksi ja kommentoitavaksi. Sähköpostiviesti. Vastaanottaja Nikulainen, J. Lähetetty 7.5.2024.

NIST. 2024. What are outliers in the data? Viitattu 19.4.2024. Saatavissa https://www.itl.nist.gov/div898/handbook/prc/section1/prc16.htm

Nawrocki, P. 2022. UX myth - flow is bad because it's too long. Viitattu 14.2.2024. Saatavissa https://uxplanet.org/ux-myth-flow-is-bad-because-its-too-long-86ecebc89c6a

Nielsen, J. 1993. Response Times: The 3 Important Limits. Viitattu 14.2.2024. Saatavissa https://www.nngroup.com/articles/response-times-3-important-limits/

Niemelä, A. 2022. Miten ja miksi käytettävyyttä tutkitaan? Johdanto käytettävyyden ja käyttäjäkokemuksen tutkimiseen. Fraktio Oy. Viitattu 11.2.2024. Saatavissa

https://www.fraktio.fi/blogi/miten-ja-miksi-kaytettavyytta-tutkitaan-johdanto-kaytettavyyden-ja-kayttajakokemuksen-tutkimiseen

Patria Group. 2024. Patria lyhyesti. Viitattu 5.3.2024. Saatavissa https://www.patriagroup.com/fi/tietoa-meista/patria-lyhyesti

Qt Group. 2024a. The future of digital experiences. Viitattu 5.3.2024. Saatavissa https://www.qt.io/group

Qt Group. 2024b. Qt Insight | Usage Intelligence for Apps & Embedded Devices. Viitattu 26.4.2024. Saatavissa https://www.qt.io/group

Riegler, A and Holzmann, C. 2018. Measuring Visual User Interface Complexity of Mobile Applications With Metrics. Oxford University Press on behalf of The British Computer Society.

Rudmin, J.W. 2010. Calculating the Exact Pooled Variance. Viitattu 16.2.2024. Saatavissa https://arxiv.org/ftp/arxiv/papers/1007/1007.1012.pdf

Rutgers University. 2023. Viitattu 11.2.2024. Saatavissa https://libguides.rutgers.edu/c.php?g=337288&p=2273210

Shanaz, K. 2023. What is Mouse Heatmap? Benefits, Tools, and Examples. Viitattu 18.2.2024. Saatavissa https://vwo.com/blog/mouse-heatmap/

Stedman, C. 2022. Definition: data analytics (DA). Techtarget. Viitattu 11.2.2024. Saatavissa https://www.techtarget.com/searchdatamanagement/definition/data-analytics

Taanila, A. 2023. Akin menetelmäblogi. Data-analytiikka Pythonilla. Viitattu 11.2.2024. Saatavissa https://tilastoapu.wordpress.com/2018/11/01/data-analytiikkaa-pythonilla/

Taboga, M. 2021. "Quantile of a probability distribution", Lectures on probability theory and mathematical statistics. Kindle Direct Publishing. Online appendix. https://www.statlect.com/fundamentals-of-probability/quantile.

Valkama, J. 2013. Kognitiivisen läpikäynnin ja simulointitestauksen hyödyntäminen ennen käytettävyystestausta. Viitattu 20.4.2024. Saatavissa https://www.cs.helsinki.fi/u/paakki/Semik13-Valkama.pdf