Tampereen yliopisto

Emma Mannila

OHJELMISTOPROJEKTIN VAATIMUSTEN KARTOITTAMISEN TEKNIIKAT JA NIIDEN KÄYTTÖ

Kandidaatintutkielma
Johtamisen ja talouden tiedekunta
Tarkastaja: Jaakko Siltaloppi
Lokakuu 2023

TIIVISTELMÄ

Emma Mannila: Ohjelmistoprojektin vaatimusten kartoittamisen tekniikat ja niiden käyttö
Kandidaatintutkielma
Tampereen yliopisto
Teknis-taloudellinen kandidaattiohjelma
Lokakuu 2023

Tämä tutkimus käsittelee ohjelmistoprojektin vaatimusten kartoittamisessa käytettyjä tekniikoita. Tutkimuksen tavoitteena on selvittää eri tekniikoille sopivia käyttötilanteita. Vaatimusten kartoittaminen on yksi osa-alue vaatimustenkäsittelyssä, joka on yksi ohjelmistoprojektin tärkeimmistä vaiheista projektin onnistumisen kannalta. Vaatimusten laiminlyöminen projektin alussa voi aiheuttaa aikataulu- ja kustannusongelmia, sekä tyytymättömyyttä valmiiseen tuotteeseen asiakkaan puolelta. Vaatimusten kartoittaminen on kriittinen vaihe oikeiden vaatimusten löytämisessä.

Tutkimuksessa käsitellään kymmentä yleisesti käytössä olevaa kartoittamistekniikkaa, esimerkiksi haastattelua, työpajoja ja prototyyppiä. Tutkimus selvittää, mitä tekniikoita vaatimusten kartoittamisessa käytetään ja minkälaisiin käyttötarkoituksiin eri kartoittamistekniikat sopivat. Tutkimus on toteutettu kirjallisuuskatsauksena, jonka alussa käsitellään, mitä kartoittamistekniikoita käytetään ohjelmistopohjaisissa projekteissa. Tekniikoista selvitetään niiden päätavoite ja käytön pääpiirteet. Toisessa osassa tutkitaan tekniikoiden ominaisuuksia ja muita seikkoja, jotka vaikuttavat tekniikan valintaan. Lopussa kootaan havainnot tekniikoista, ja selvitetään missä tilanteissa tekniikoita kannattaa käyttää ja mitä keskeisiä rajoitteita käytölle on.

Tekniikoille löytyy selkeitä raameja eri käyttötilanteisiin. Käyttöön vaikuttaa muun muassa tekniikan ominaisuudet, kartoittajan kokemus ja projektiympäristö. Tutkimuksen perusteella esimerkiksi kartoittamisen alussa kannattaa käyttää aivoriihtä, nopeassa aikataulussa työpajat ovat sopiva tekniikka ja haastattelun avulla taas saadaan laajaa ja yksityiskohtaista tietoa. Haastattelut, työpajat ja aivoriihet ovat tutkimuksen monikäyttöisimpiä tekniikoita, kun taas muille tekniikoille löytyy tarkempia käyttökohteita. Rajoitteina tekniikoiden käytölle ovat esimerkiksi aikatauluhaasteet, kartoittajan osaaminen ja asiakkaan tekninen ymmärrys. Jokaiseen projektiin on löydettävä siihen sopivat tekniikat, ja tämä tutkimus antaa raameja päätöksentekoon.

Avainsanat: ohjelmistoprojekti, vaatimusten kartoittaminen, kartoittamistekniikat

SISÄLLYSLUETTELO

1. JOHDANTO

Ohjelmistokehitys toteutetaan yleensä projektimuotoisesti. Ohjelmistoprojektin toteutukseen on olemassa monia erilaisia projektimalleja, kuten vesiputousmalli ja ketterät mallit. Mallista riippumatta ohjelmisto toteutetaan lähtökohtaisesti asiakkaalta tulevien vaatimusten perusteella. Vaatimusten kerääminen voi olla pääosin projektin alussa tehtävä aktiviteetti tai sitä voi tapahtua iteratiivisesti projektin aikana. Vaatimukset pyritään määrittelemään projektin alussa mahdollisimman hyvin, mutta usein vaatimuksiin tulee myös muutoksia projektin aikana. (Haikala & Mikkonen 2011)

Vaatimusten kartoittaminen (engl. requirements elicitation) on joukko aktiviteetteja, joiden avulla saadaan selville systeemin ominaisuudet, joita sidosryhmät edellyttävät, sekä systeemin toimintaympäristön vaatimukset (Kotonya & Sommerville 1998). Tässä työssä systeemeistä tarkastelussa on ohjelmistot. Esimerkiksi Wagner et al. (2019) toteuttaman kyselytutkimuksen mukaan haastattelut, fasilitoidut tapaamiset (muun muassa työpajat) ja prototyypit ovat käytetyimpiä ohjelmistoprojektien kartoittamistekniikoita tällä hetkellä, minkä lisäksi käyttötapauksia ja tarkkailua käytetään laajasti. Tekniikoille ei ole yhtä totuutta siitä, milloin ja miten niitä olisi parasta käyttää, vaan valinta riippuu aina projektin yksilöllisestä ympäristöstä (Sharma & Pandey 2014). Kirjallisuudessa on esitetty erilaisia malleja kartoittamisen toteuttamiseksi ja sopivien tekniikoiden valitsemiseksi (Hickey & Davis 2004; Muqeem et al. 2022). Sopivien tekniikoiden valitsemiseksi on kuitenkin omattava ymmärrystä yksittäisistä tekniikoista ja niiden käytöstä, joihin tämä työ syventyy.

Vaatimusten kartoittaminen on osa isompaa vaatimustenkäsittelyn (engl. requirements engineering) prosessia (Bourque & Fairley 2014; Kotonya & Sommerville 1998; Pressman 2005). Vaatimustenkäsittely on prosessi, jonka tavoitteena on kerätä, vahvistaa ja ylläpitää ohjelmiston vaatimuksia (Kotonya & Sommerville 1998). Vaatimusten käsittelyn ja kartoittamisen prosessit esitellään tarkemmin luvussa 2.

Ohjelmistoprojektien epäonnistumiset ovat olleet esillä julkisessa keskustelussa lähivuosina. Haikalan ja Mikkosen (2011) mukaan suuri osa näistä epäonnistumisista voidaan johtaa heikkoon vaatimustenkäsittelyyn. Myös Hussain et al. (2016) tutkimuksen perusteella hyvä vaatimustenkäsittely on pohja onnistuneelle ohjelmistoprojektille. Tutkimuk-

sen perusteella yksi suurimmista haasteista vaatimustenkäsittelyprosessissa ovat epätäydelliset ja piilevät vaatimukset (Fernández et al. 2017), minkä vuoksi vaatimusten kartoittaminen on kriittinen vaihe prosessissa.

1.1 Tutkimuksen tavoite ja tutkimusongelma

Työn tavoitteena on tutkia, mitä tekniikoita vaatimusten kartoittamisessa käytetään, mitä ominaisuuksia näillä tekniikoilla on ja minkälaisissa tilanteissa niitä on sopiva käyttää. Työssä tarkastellaan tekniikoiden vahvuuksia ja heikkouksia ja niiden perusteella tutkitaan, miten tekniikoita kannattaa käyttää. Tutkimuskysymyksiksi valittiin seuraavat:

(1) Mitä tekniikoita vaatimusten kartoittamisessa käytetään?

(2) Minkälaisiin käyttötarkoituksiin eri kartoittamistekniikat sopivat?

Vaatimusten kartoittamisen prosessiin kuuluu kartoittamistekniikoiden käyttämisen lisäksi muun muassa tavoitteiden asettaminen ja taustatyön tekeminen, jonka aikana projektin toteuttajat hankkivat tietoa ja ymmärrystä asiakkaan organisaatiosta ja alasta (Kotonya & Sommerville 1998). Työn pääpaino on tekniikoiden tarkastelussa ja käyttötilanteiden tutkimisessa, ja prosessia sivutaan teoriaosassa hieman.

Työn toisen luvun alussa käsitellään vaatimuksia ja vaatimusten käsittely- ja kartoittamisprosessia yleisesti. Sen jälkeen toisessa luvussa esitellään kartoittamistekniikoita ja niiden perustietoa. Kolmannessa luvussa keskitytään tekniikoiden ominaisuuksiin ja käyttöön. Luvussa käsitellään sitä, miten eri tekniikoiden ominaisuudet sopivat eri tilanteisiin esimerkiksi projektin koon sekä kartoitettavien vaatimusten ominaisuuksien näkökulmasta. Neljännessä luvussa vertaillaan tuloksia ja pohditaan niiden merkitystä. Myös tulevan tutkimuksen mahdollisuuksia pohditaan.

1.2 Tutkimusmetodologia

Tutkimus on toteutettu kirjallisuuskatsauksena. Aiheen rajauksen jälkeen valittiin sopivat hakusanat. Oleellisimmat termit kartoittamistekniikoita varten ovat ”requirements elicitation” ja technique, joille valittiin synonyymeiksi ”requirements gathering” ja method. Näiden perusteella luotiin sopivat hakulausekkeet. Myöhemmin kartoittamistekniikoista haettiin tietoa käyttämällä hakusanoja ”requirements elicitation”, ”requirements gathering” ja tekniikan nimi, esimerkiksi ”workshop”. Tiedonhaku toteutettiin Web of Science (WOB) ja Scopus -tietokannoista. Tuloksista luettiin otsikot, joiden perusteella relevanteista artikkeleista luettiin tiivistelmät. Tiivistelmien perusteella valittiin artikkelit syvempään tarkasteluun. Artikkeleiden valinnan kriteereinä olivat se, että ne käsittelevät vaatimusten kartoittamista ohjelmistoprojekteissa ja käsiteltävä aihe ei ole tietty yksittäinen ohjelmisto

tai sovellusala. Lopullisiksi lähteiksi valikoitui 31 artikkelia ja teosta. Taulukossa 1 on eritelty hakulausekkeet, rajaukset ja tulosten määrä. Koska tiedonhaulla ei löytynyt tarpeeksi perustietoa tekniikoiden käytöstä, otettiin avuksi myös Schlosser et al. (2006) helmenkasvatusmenetelmä. Tiedonhaun avulla löytyi helmiartikkeleita, joiden avulla löytyi sopivia artikkeleita ja teoksia kartoittamistekniikoista. Helmenkasvatusmenetelmällä löytyi 8 lähdettä. Liitteessä A on kuvattu helmenkasvatusprosessi.

Taulukko 1 Tiedonhakulausekkeet

Hakulauseke	Rajaukset	Tulosten määrä
(”requirements elicitation” OR ”requirements gathering”) AND (technique* OR method*)	Vuosirajaus 2019-2023 Artikkelit	WOB: 135 Scopus 213
("requirements elicitation" OR "requirements gathering") AND workshop*	Artikkelit	WOB: 32 Scopus: 26
("requirements elicitation" OR "requirements gathering") AND brainstorming	Artikkelit	WOB: 8 Scopus: 13
("requirements elicitation" OR "requirements gathering") AND cognitive	Artikkelit	WOB: 21 Scopus: 28
("requirements elicitation" OR "requirements gathering") AND “field study”	Artikkelit	WOB: 3 Scopus: 18

2. VAATIMUSTEN KARTOITTAMINEN

Haikalan ja Mikkosen (2011) mukaan ohjelmistoprojektin vaiheisiin kuuluu projektimallista riippumatta yleensä määrittely, suunnittelu, ohjelmointi, testaus, käyttöönotto ja ylläpito. Näiden vaiheiden toteutus riippuu projektimallista, esimerkiksi vesiputousmallissa vaiheet ovat tavallisessa järjestyksessä, kun taas ketterissä menetelmissä vaiheet ovat enemmän päällekkäisiä ja toiminta on iteratiivisempaa. Määrittelyvaiheessa kerätään asiakkaan vaatimuksia ja luodaan niistä dokumentaatio ohjelmiston ominaisuuksista. Vaatimustenkäsittely ja kartoittaminen kuuluvat siten määrittelyvaiheeseen. (Haikala & Mikkonen 2011)

Kuten johdannossa on mainittu, vaatimusten kartoittaminen kuuluu vaatimustenkäsittelyn prosessiin. Eri lähteissä vaatimustenkäsittelyprosessi kuvataan eri tavoilla ja prosessiin kuuluu eri aktiviteetteja. Kotonya ja Sommerville (1998) määrittelevät prosessin koostuvan vaatimusten kartoittamisesta, analyysistä ja neuvottelusta, dokumentaatiosta ja vahvistamisesta. Pressmanin (2005) mukaan prosessiin kuuluu aloitus, kartoittaminen, säätäminen, neuvottelu, spesifikaatio, vahvistaminen ja ylläpito. Bourque ja Fairley (2014) listaavat prosessin aktiviteeteiksi kartoittamisen, analyysin, spesifikaation ja vahvistamisen. Kaikissa tapauksissa kartoittaminen kuitenkin kuuluu prosessin alkuvaiheeseen.

2.1 Vaatimukset ja kartoittamisprosessi

Haikala ja Mikkonen (2011, s. 61) määrittelevät vaatimuksen seuraavasti: ”Vaatimus (requirement) on jotain, mitä tuotteella pystyy tekemään, tai (laatu-) ominaisuus, joka tuotteella tulee olla.” Haikalan ja Mikkosen (2011) mukaan vaatimukset voivat olla toiminnallisia (engl. functional), ei-toiminnallisia (engl. non-functional) tai reunaehtoja (engl. constraints). Ali Ramdhani et al. (2018), Bourque ja Fairley (2014) ja Kotonya ja Sommerville (1998) jakavat vaatimukset toiminnallisiin ja ei-toiminnallisiin, ja näissä tapauksissa Haikalan ja Mikkosen (2011) reunaehdot kuuluvat ei-toiminnallisten vaatimusten määritelmään. Toiminnalliset vaatimukset ovat kuvauksia toiminnoista, joita ohjelmiston täytyy suorittaa. Ei-toiminnalliset vaatimukset rajaavat ohjelmiston toteutusta, ja ne voivat liittyä esimerkiksi suorituskykyyn, ylläpitoon, turvallisuuteen, luotettavuuteen tai yhteentoimivuuteen. (Bourque & Fairley 2014) Reunaehdot voivat rajata esimerkiksi käytetyn ohjelmointikielen (Haikala & Mikkonen 2011).

Bourquen ja Fairleyn (2014) mukaan vaatimusten kartoittamisprosessi alkaa projektin sisällön määrittelyllä, johon kuuluu yleinen kuvaus toteutettavasta ohjelmistosta ja ohjelmiston tarkoituksen määrittely. Tämä luo raamit kartoitettaville vaatimuksille, jotta vaatimukset pysyvät relevantteina. Kotonya ja Sommerville (1998) aloittavat vaatimusten kartoittamisen samalla vaiheella. Tämän jälkeen Kotonyan ja Sommervillen (1998) mukaan tehdään laadittavan ohjelmiston taustaselvitys, johon kuuluu ohjelmiston käyttöönottavan organisaation tutkiminen, toimialaselvitys ja olemassa olevien ohjelmistojen selvitys. Bourquen ja Fairleyn (2014) mukaan sisällönmäärittelyn jälkeen tunnistetaan ja arvioidaan vaatimusten kaikki mahdolliset lähteet. Lähteiden tunnistaminen kuuluu myös Kotonyan ja Sommervillen (1998) ja Pohl:n (2010) kartoittamisprosessiin. Erilaisia lähteitä ovat esimerkiksi yrityksen tavoitteet, sidosryhmät, yrityksen sisäiset säännöt ja operatiivinen ympäristö. Lähteiden tunnistamisen jälkeen alkaa kartoittaminen, jossa käytetään kartoittamistekniikoita. (Bourque & Fairley 2014; Kotonya & Sommerville 1998; Pohl 2010) Taulukossa 2 on eritelty kartoittamisprosessin aktiviteetit eri lähteiden perusteella.

Taulukko 2. *Kartoittamisprosessin aktiviteetit*

Kirjallisuuslähde	Sisällönmäärittely	Taustatyö	Lähteiden tunnistus	Vaatimusten kartoittaminen
Bouque & Fairley (2014)	x		x	x
Kotonya & Sommerville (1998)	x	x	x	x
Pohl (2010)			x	x

2.2 Kartoittamistekniikat

Työssä tutkittavien kartoittamistekniikoiden valitsemisessa on käytetty neljää eri lähdettä, joissa esiintyy tekniikoiden kategorisointi (Batra & Bhatnagar 2017, Batra & Bhatnagar 2019 mukaan), kirjallisuuskatsaus yleisimmistä tekniikoista (Aflen & Vasques 2021), empiirinen tutkimus yleisimmistä tekniikoista (Wagner et al. 2019) ja kartoittamistekniikoiden haasteiden tutkimus (Sharma & Pandey 2014). Batran ja Bhatnagarin (2017) mukaan kartoittamistekniikat voidaan jakaa neljään kategoriaan seuraavasti: keskustelevat tekniikat (engl. conversational), yhteistekniikat (engl. collaborative), kognitiiviset tekniikat (engl. cognitive) ja havaintotekniikat (engl. observational) (katso Batra & Bhatnagar 2019). Sama jaottelu on käytössä myös tässä työssä. Batra ja Bhatnagar (2019) ovat tunnistaneet kategorioihin kuuluvia tekniikoita, jotka ovat eritelty taulukossa 3. Aflenin ja Vasquesin (2021) toteuttamassa kirjallisuuskatsauksessa on tunnistettu 40 yleisintä kirjallisuudessa esiintyvää kartoittamistekniikkaa. Artikkelissa tutkittiin tarkemmin 10:tä tekniikkaa, joista on tähän työhön valittu 7 tekniikkaa sen perusteella, että ne

ovat mainittu myös muissa edellä mainituissa lähteissä. Wagner et al. (2019) tutkivat vaatimustenkäsittelyn nykytilaa yrityksissä kyselytutkimuksella, jonka avulla saatiin selville käytännön käytetyimpiä kartoittamistekniikoita. Eri tekniikoiden haasteita on tutkittu Sharman ja Pandeyn (2014) artikkelissa. Taulukkoon 3 on eritelty edellä esitetyissä artikkeleissa käytettyjä tekniikoita, joita tutkitaan tässä työssä.

Taulukko 3. *Työssä tutkitut kartoittamistekniikat*

Tekniikka ja kategoria	Sharma & Pandey (2014)	Batra & Bhatnagar (2019)	Wagner et al. (2019)	Aflen & Vasques (2021)
Keskustelevat				
Haastattelut	x	x	x	x
Työpajat	x	x	x	x
Aivoriihet	x	x		x
Kyselyt		x		x
Yhteistekniikat				
Prototyypit	x		x	x
Käyttötapaukset		x	x	x
Kognitiiviset				
Porrastus	x	x		
Korttilajittelu	x	x		
Repertory grid	x	x		
Havaintotekniikat				
Tarkkailu	x	x	x	x

2.2.1 Keskustelevat tekniikat

Keskustelevat tekniikat ovat perinteisimpiä kartoittamistekniikoita, ja niiden tarkoituksena on, että sidosryhmät pukevat vaatimuksia sanoihin ja kommunikoivat niitä vaatimusten kartoittajalle (Sharma & Pandey 2014). Tässä työssä keskusteleviin tekniikoihin kuuluu haastattelu, työpajat, aivoriihet ja kyselyt.

Haastattelu on yksi käytetyimmistä tekniikoista vaatimusten kartoittamisessa. Sen tavoitteena on kerätä syvällistä tietoa kahden tai useamman ihmisen välisessä vuorovaikutuksessa. Haastattelu voi olla strukturoitu, strukturoimaton tai semi-strukturoitu. Strukturoitu haastattelu on raameiltaan tiukin, ja siinä kysymykset ovat useimmiten suljettuja ja haastattelija pitäytyy valmiiksi valituissa kysymyksissä. Strukturoimattomassa haastattelussa kysymykset ovat avoimia ja haastateltava voi syventyä yhteen aiheeseen haluamansa mukaan. Semi-strukturoidussa haastattelussa yhdistyvät edellä mainitut haastattelutyypit. Siinä voi olla avoimia ja suljettuja kysymyksiä, ja haastattelija voi syventää keskustelua eri aiheista tarpeen mukaan. (Baxter & Courage 2005)

Työpajat (engl. workshop) ovat keskustelutilaisuuksia, joissa ohjelmiston kehittäjät ja sidosryhmät keskustelevat vaatimuksista. Työpajan tavoitteena on kehittää vaatimuksia yhdessä ja löytää yhteisymmärrys eri sidosryhmien välillä. Tilaisuuden järjestämiseen kuuluu tavoitteiden asettaminen ja sopivien sidosryhmien edustajien valitseminen, jotta saadaan kartoitettua oikeita vaatimuksia. Riippuen osallistumismäärästä, työpajassa voidaan työskennellä yhdessä ryhmässä tai useammassa pienryhmässä. (Pohl 2010)

Aivoriihet ovat vapaamuotoisia keskustelutilaisuuksia, joiden tavoitteena on kerätä eri sidosryhmiltä ideoita ohjelmistoa varten (Yousuf & Asger 2015). Aivoriihi on innovatiivisuutta ja luovuutta hyödyntävä tekniikka (Pohl 2010). Tilaisuuden toteutus voidaan jakaa kahteen osaan, ideoimiseen ja ideoiden arviointiin (Paetsch et al. 2003). Ideoimisen aikana jokaiselle osallistujalle annetaan mahdollisuus jakaa omia ideoita tilaisuuden teemaan liittyen, ja ideat dokumentoidaan. Arviointivaiheessa epäsopivat ideat poistetaan ja sopivat ideat priorisoidaan. (Yousuf & Asger 2015).

Kysely on tarkkaan suunniteltu joukko kysymyksiä, jonka avulla voidaan kerätä tietoa suurelta joukolta ihmisiä (Baxter & Courage 2005). Kyselyiden tavoite on kerätä paljon tarkkaa tietoa (asenteita, faktoja, käyttäytymistä) tuotteeseen liittyen. Kyselyn laatiminen vaatii tarkkaa ymmärrystä tuotteesta ja tietoa kyselyiden laatimisesta sosiaalisesta, taloudellisesta ja psykologisesta näkökulmasta. Kysymysten valinta ja muotoilu ovat hyvin tärkeitä, jotta vastaajat osaavat vastata siihen, mitä kysytään. (Belani et al. 2005) Pohl (2010) suosittelee liittämään kyselyyn yhteyshenkilön, johon voi olla yhteydessä, jos vastaaja ei osaa vastata kaikkiin kysymyksiin.

2.2.2 Yhteistekniikat

Tässä tutkimuksessa yhteistekniikoihin lukeutuu prototyypit ja käyttötapaukset. Molemmat tekniikan perustuvat haastatteluun, jonka apuvälineenä käytetään työkalua, prototyyppiä tai käyttötapauskuvausta.

Prototyyppi on varhainen versio ohjelmistosta (Kotonya & Sommerville 1998), jonka tavoitteena on mahdollistaa sidosryhmille osittainen ohjelmiston tarkastelu käytännössä (Pohl 2010). Sidosryhmän edustaja voi verrata omia odotuksiaan toteutettuun prototyyppiin, minkä avulla voidaan saada palautetta ja löytää uusia vaatimuksia (Pohl 2010). Prototyyppi voi olla kertakäyttöinen tai kehityksellinen (Paetsch et al. 2003; Kotonya & Sommerville 1998). Kertakäyttöinen prototyyppi on yksinkertaisempi ja auttaa ymmärtämään monimutkaisia vaatimuksia, kun taas kehityksellistä prototyyppiä voidaan jatkokehittää, jolloin siitä usein tulee osa lopullista systeemiä (Paetsch et al. 2003). Kotonyan ja Sommervillen (1998) mukaan vaatimusten kartoittamisessa käytetyt prototyypit ovat usein kertakäyttöisiä, koska ne vaativat vähemmän aikaa toteutukseen ja niihin voi toteuttaa

vain niitä osa-alueita, joiden vaatimuksissa on vielä epäselvyyksiä. Kertakäyttöiset prototyypit voivat olla toiminnallisia ohjelmistoja, malleja ilman toiminnallisuuksia tai paperiprototyyppejä (Pohl 2010).

Käyttötapaukset ovat kuvauksia käyttäjän ja ohjelmiston välisestä vuorovaikutuksesta. Käyttötapaus kuvaa ohjelmiston käyttötilannetta vaihe vaiheelta, ja sen avulla voidaan kartoittaa funktionaalisia vaatimuksia aikaisessa kehitysvaiheessa. (Paetsch et al. 2003) Käyttötapauksen kehittäminen vaatii yhteistyötä ohjelmiston kehittäjien ja loppukäyttäjien välillä (Kotonya & Sommerville 1998). Käyttötapaus sisältää koko vuorovaikutuksen prosessin, johon kuuluu esimerkiksi lähtötilanne, tapahtumien kulku, samanaikaiset tapahtumat ja lopputilanne (Kotonya & Sommerville 1998; Yousuf & Asger 2015). Käyttötapaukset voidaan kuvata luonnollisella kielellä tai graafisena esityksenä (Kotonya & Sommerville 1998).

2.2.3 Kognitiiviset tekniikat

Kognitiiviset tekniikat hyödyntävät kartoittajan kokemusta ja jo olemassa olevaa dokumentaatiota järjestelmästä, ja tekniikoiden avulla voidaan kartoittaa vaatimuksia hyödyntäen asiakkaan sisäisiä oletuksia (Batra & Bhatnagar 2019).

Porrastus (engl. laddering) on haastattelutekniikka, joka perustuu kysymysten hierarkkiseen järjestämiseen. Aluksi käsitellään ohjelmiston päävaatimuksia, joiden avulla voidaan käsitellä tarkempia vaatimuksia syvemmältä tasolta. Porrastustekniikan tarkoituksena on priorisoida sidosryhmien vaatimuksia. Hierarkkinen malli tekee tekniikasta melko joustamattoman, koska järjestettyihin kysymyksiin ja vaatimuksiin on vaikeaa lisätä uutta. (Yousuf & Asger 2015).

Korttilajittelu (engl. card sorting) on tekniikka, jonka avulla pyritään havainnollistamaan käyttäjän näkemystä tuotteen rakenteesta ja sisällöstä. Tuotteessa olevia tehtäviä tai informaatiopalikoita kirjoitetaan korteille, jonka jälkeen sidosryhmät lajittelevat kortteja kategorioihin eri kriteerien perusteella. Tekniikan avulla saadaan selville, mitkä osat kuuluvat tuotteessa sidosryhmien mielestä yhteen. Korttilajittelun tavoitteena on selvittää käyttäjien mentaalinen malli tuotteesta, jolloin lopputuotteen käyttö on mahdollisimman vaivatonta ja käyttäjä osaa navigoida järjestelmän sisällä. (Baxter & Courage 2005) Tilaisuuden aikana lajittelu eri kategorioihin voidaan toteuttaa useamman kerran eri kriteerien mukaisesti. Kriteerit ja kategoriat voivat olla etukäteen valitut kartoittajan toimesta, tai sidosryhmät voivat valita kriteerit ja kategoriat itse jokaisen kierroksen aikana. (Rugg & McGeorge 1997)

Repertory grid on George Kellyn (1955) kehittämä haastattelutekniikka, joka perustuu hänen ”henkilökohtaisen konstruktion teoriaan” (engl. Personal Construct Theory). Teorian mukaan ihmiset käsittelevät asioita ja tapahtumia mielessään luomalla konstruktioita. (katso Dey & Lee 2015) Tekniikka perustuu ruudukkoon, joka koostuu elementeistä ja konstruktioista. Elementit ovat asioita, joita voidaan käsitteellistää konstruktioiden avulla. Konstruktiot ovat binäärisiä näkökulmia, esimerkiksi kiltti vs. ilkeä. Elementit sijoittuvat ruudukon sarakkeisiin ja konstruktiot riveille. Elementtejä voi näin arvioida konstruktioiden avulla. (George Kelly 1955, katso Fransella et al. 2004) Tekniikan tarkoituksena on arvioida ja eritellä vaatimuksia. Ruudukon täyttämisessä käytetään usein apuna semi-strukturoitua haastattelua. (Yousuf & Asger 2015)

2.2.4 Havaintotekniikat

Tarkkailu (engl. observation) on osa kenttätutkimusmetodia, jonka tarkoituksena on kerätä laadullista dataa ohjelmiston käyttäjän arjen aktiviteeteista (Bly 1997). Tarkkailun aikana vaatimusten kartoittaja havainnoi käyttäjää työympäristössään (Yousuf & Asger 2015). Tarkkailu voi tapahtua suoraan, eli tarkkailija on paikan päällä, tai epäsuoraan, jolloin tarkkailu tapahtuu esimerkiksi videon välityksellä (Paetsch et al. 2003). Tarkkailun avulla saatu data on laadullista, jonka vuoksi tarkkailtavien käyttäjien määrä on yleensä pieni (Kujala 2008).

Tarkkailun avulla voi saada selville muun muassa minkälaisessa kontekstissa ohjelmistoa käytetään, mitä aktiviteetteja ohjelmiston täytyy tukea ja mitä uusia mahdollisuuksia ohjelmisto voi tuoda työtehtäviin. Ohjelmiston kehittäjällä voi olla myös omia ajatuksia vaatimuksista, joiden oikeellisuudesta voi saada varmistuksen tarkkailun avulla. (Bly 1997) Tarkkailun avulla pystytään näkemään, mitä työtehtävissä oikeasti tapahtuu, koska työntekijät saattavat puhuessaan yksinkertaistaa aktiviteetteja (Paetsch et al. 2003). Sharman ja Pandeyn (2014) mukaan tarkkailu tuo esille implisiittiä vaatimuksia sekä ”laajaa tietoa sidosryhmistä, heidän taustastaan, toimintamalleista, työtekniikoista, flow:sta ja muista päivittäiseen työhön liittyvistä aiheista”.

2.2.5 Yhteenveto tekniikoista

Keskustelutekniikoiden avulla vaatimuksia kerätään sanallisesti kahden tai useamman ihmisen välillä. Haastattelu voidaan toteuttaa monella eri tavalla tarpeen mukaan. Työpajat ja aivoriihet ovat ryhmäkeskusteluja, mutta työpajojen rakenne on strukturoidumpi, kun taas aivoriihissä keskustelu on vapaampaa ja innovatiivisempaa. Kyselyn avulla vaatimusten kartoittaminen tapahtuu ilman suoraa vuorovaikutustilannetta. Yhteistekni-

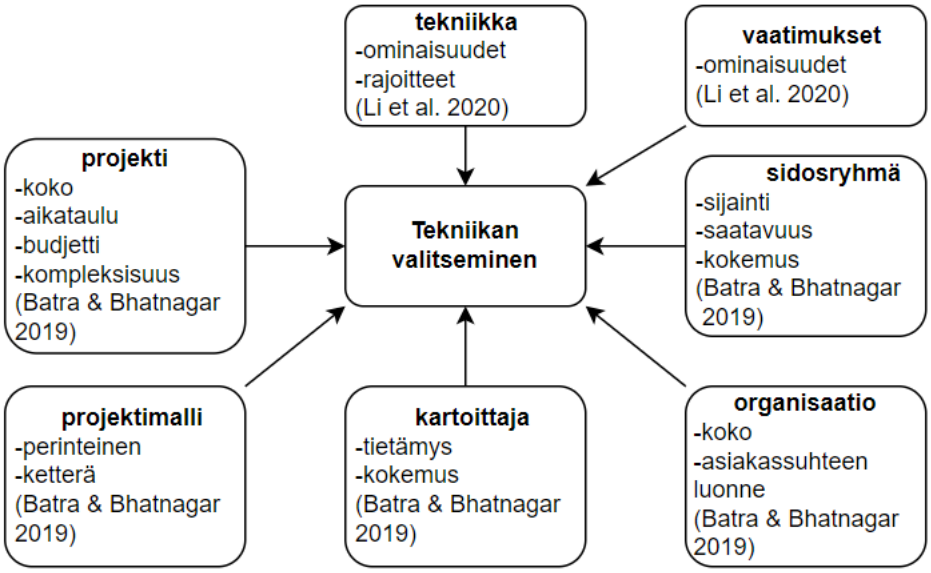
koita käytettäessä vaatimusten kartoittamisessa käytetään jotakin apuvälinettä. Prototyyppi on teknisempi keino ja sen avulla on mahdollista kehittää ohjelmistoa jo käytännössä. Käyttötapaukset hyödyntävät käyttäjän kokemusta järjestelmän käytöstä. Kognitiiviset tekniikat hyödyntävät käyttäjän sisäisiä malleja ja ajatuksia järjestelmään liittyen. Porrastamisen avulla voidaan priorisoida vaatimuksia hierarkkisesti, korttilajittelulla havainnollistetaan järjestelmän rakennetta ja repertory grid -tekniikalla voidaan arvioida ja vertailla vaatimuksia. Havainnointitekniikat, eli tässä tutkimuksessa tarkkailu, eivät vaadi suoraa kontaktia asiakkaaseen. Tarkkailussa havainnoidaan järjestelmän käyttöä käytännössä.

3. KARTOITTAMISTEKNIIKOIDEN KÄYTTÖ

3.1 Tekniikan valitsemiseen vaikuttavat tekijät

Batra ja Bhatnagar (2019) esittävät kartoittamistekniikan valitsemiseen mallin, jonka mukaan valintaan vaikuttavat projektin, sidosryhmän, kartoittajan ja organisaation ominaisuudet sekä ohjelmistokehitysprosessin muoto. Projektin ominaisuuksia, jotka voivat vaikuttaa valintaan ovat esimerkiksi projektin koko, aikataulurajoitteet, budjetti ja kompleksisuus eli monimutkaisuus. Sidosryhmästä vaikuttavia tekijöitä ovat esimerkiksi maantieteellinen sijainti, saatavuus ja kokemus. (Batra & Bhatnagar 2019) Esimerkiksi myös Aljuhani et al. (2021) toteuttaman tutkimuksen mukaan avainsidosryhmien saatavuus on yksi suurimmista tekniikan valitsemiseen vaikuttavista kriteereistä. Muita tutkimuksen perusteella merkittävimpiä valintaan vaikuttavia kriteerejä ovat resurssien saatavuus, käyttäjän kuvaileva luonne (engl. expressiveness) ja käyttäjän motivaatio ja yhteistyöhalukkuus. Kartoittajan ominaisuuksia ovat esimerkiksi alan tietämys, kokemus ja ongelmakentän tietämys. Organisaation kontekstissa tekniikan valintaan voi vaikuttaa esimerkiksi organisaation koko ja asiakassuhteen luonne. Ohjelmistoprosessi taas voi olla esimerkiksi perinteinen vesiputousmalli tai ketterä malli, mikä vaikuttaa myös tekniikan valintaan. (Batra & Bhatnagar 2019) Esimerkiksi haastattelut, prototyypit ja aivoriihet ovat osittain käytettyjä myös ketterän kehityksen menetelmissä, kun taas havainnointi ei ole siinä käytetty (Younas et al. 2017).

Li et al. (2020) mukaan sopivan tekniikan valitsemiseen vaikuttaa projektiympäristön lisäksi kerättävät vaatimukset ja tekniikan ominaisuudet. Seuraavassa kappaleessa käsitellään tekniikan valintaa juuri sen ominaisuuksien näkökulmasta. Kuvaan 1 on havainnollistettu kaikki edellä mainitut tekniikan valitsemiseen vaikuttavat tekijät.



Kuva 1 Tekniikan valitsemiseen vaikuttavat tekijät

3.2 Kartoittamistekniikat

Seuraavissa alakappaleissa käsitellään kappaleessa 2 esiteltyjen tekniikoiden käyttöä. Tekniikoista käsitellään niiden vahvuuksia ja heikkouksia, rajoitteita, ajankäyttöä ja minkälaisiin projekteihin ne soveltuvat parhaiten.

3.2.1 Keskustelevat tekniikat

Haastattelun yksi suurimmista vahvuuksista on sen tuottama runsas, rikas ja yksityiskohtainen data. Haastattelun avulla voi saada hyvän kokonaiskäsityksen kehitettävästä ohjelmistosta. (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015) Haastattelua voi käyttää myös alustavana tekniikkana, jonka tulosten perusteella voi ottaa käyttöön muita tekniikoita, esimerkiksi kyselyn (Shams-Ul-Arif et al. 2009). Haastattelun heikkoutena on se, että sen avulla saa tietoa vain pieneltä ihmisjoukolta kerrallaan, jonka vuoksi se on hidas ja vaatii resursseja (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015). Haastattelun aikana on tärkeää, että haastateltava tuntee olonsa mukavaksi. Yousufin ja Asgerin (2015) mukaan haastateltava ei välttämättä kerro kaikkea relevanttia informaatiota eikä rohkeasti uusia ideoita, jos hän ei tunne oloaan mukavaksi.

Guptan ja Deramanin (2019) mukaan haastattelu sopii monipuolisesti erilaisiin projekteihin. Strukturoimaton haastattelu on parempi vaihtoehto pienempiin projekteihin, mutta se toimii myös hyvin monimutkaisissa projekteissa. Strukturoitu haastattelu taas sopii paremmin suurempiin projekteihin, joissa ei ole niin paljoa monimutkaisuutta. (Gupta & Deraman 2019)

Rueda et al. (2020) tutkimuksen mukaan strukturoimaton haastattelu on nopeampi ja helpompi järjestää, mutta se tuottaa määrällisesti vähemmän relevantteja vaatimuksia muihin haastatteluihin verrattuna, minkä vuoksi Rueda et al. suosittelevat strukturoimattoman haastattelun käyttämistä silloin, kun valmisteluaikaa on vähän, kartoittajalla on vähän kokemusta tekniikasta ja selvitettävä ongelma ei ole suuri. Toisaalta Guptan ja Deramanin (2019) mukaan molemmat yllä mainituista haastattelutyypeistä vaativat kartoittajalta jonkin verran kokemusta.

Työpajoissa on mahdollista ratkaista suuria ja monimutkaisia ongelmia. Tekniikan avulla kerättävät vaatimukset ovat usein myös melko kokonaisia ja muuttumattomia. Tilaisuudessa kerätyt vaatimukset kattavat yleensä koko tilaisuuden teemaan liittyvän ongelma-alueen. (Shams-Ul-Arif et al. 2009). Työpajan avulla saadaan luotua yhteisymmärrystä sidosryhmien välille (Yousuf & Asger 2015). Mich et al. (2023) tutkimuksen perusteella ryhmätekniikoiden, kuten työpajojen ja aivoriihien avulla voidaan saada parempia ratkaisuja ja enemmän näkökulmia vaatimuksiin. Epäjohdonmukaisuus ja toistuvuus vaatimuksissa taas vähenee ryhmätyön avulla. (Mich et al. 2023) Työpajan avulla saa myös kerättyä vaatimuksia nopeammin kuin esimerkiksi haastattelussa, jonka avulla voidaan saada vaatimuksia vain yhdeltä henkilöltä kerrallaan (Yousuf & Asger 2015).

Työpaja voi aiheuttaa haasteita usean osallistujan vuoksi. Koska työpajaan täytyy osallistua useita sidosryhmien edustajia, kaikkien saaminen paikalle voi olla vaikeaa. Jos paikalle on valittu liikaa keskustelijoita, työpaja voi edetä hitaasti. Toisaalta liian pienellä määrällä osallistujia vaatimukset voivat jäädä vajaiksi, eikä kaikkien sidosryhmien tarpeet tule esille. (Yousuf & Asger 2015) Shams-Ul-Arif et al. (2009) ja Batran ja Bhatnagarin (2019) mukaan työpajat eivät sovi pieniin projekteihin.

Aivoriihen suuri vahvuus on sen innovatiivisuus. Aivoriihet edistävät vapaata ja innovatiivista ajattelua, jonka avulla voidaan löytää pääratkaisut ohjelmiston toteutukselle. (Shams-Ul-Arif et al. 2009) Yousufin ja Asgerin (2015) mukaan aivoriihet ovat ymmärrettäviä ja helppoja toteuttaa. Okesola et al. (2019) mukaan on olemassa valmiita pohjia, joiden avulla tilaisuuden voi järjestää. Aivoriiheen osallistuakseen sidosryhmän edustajan ei tarvitse olla aiheen asiantuntija (Yousuf & Asger 2015). Carrizo et al. (2014) mukaan aivoriihi sopii myös tilanteisiin, joissa sidosryhmien kesken ei ole yhteisymmärrystä vaatimuksista. Myös Yousufin ja Asgerin (2015) mukaan tekniikka toimii sidosryhmien välisten eturistiriitojen selvittämisessä. Batra ja Bhatnagar (2019) sitä vastoin ovat sitä mieltä, että tekniikka ei sovi käytettäväksi, kun sidosryhmillä on erilaiset tarpeet.

Aivoriihen avulla ei pysty ratkaisemaan suuria toteutukseen liittyviä ongelmia (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015). Suuri määrä ideoita ei myöskään takaa, että

ideat sisältävät laadukasta ja relevanttia informaatiota (Yousuf & Asgar 2015). Carrizo et al. (2014) suosittelee, että aivoriihen toteuttaa kartoittaja, jolla on osaamista ja kokemusta kartoittamisesta yleisesti, sekä aivoriihestä tekniikkana.

Aivoriihi sopii tekniikkana parhaiten suuriin ja monimutkaisiin projekteihin. Tekniikan toteuttamisessa kestää suhteellisen kauan, ja se sopii parhaiten käytettäväksi vaatimusten kartoittamisen alussa. Aivoriihen avulla saatu tieto on runsasta ja relevanttia. (Gupta & Deraman 2019)

Kysely on Carrizo et al. (2014) mukaan joustava vastaajien sijainnin ja ajankäytön puolesta. Kysely voidaan jakaa suurelle massalle ihmisiä, ja vastaaja voi vastata kyselyyn missä vain, eikä vastaaminen välttämättä vaadi suuresti aikaa. Vastaajan ei myöskään tarvitse omata teknistä osaamista ohjelmistosta. (Carrizo et al. 2014) Kyselyn avulla voi kerätä tilastollista tietoa kartoittajan ja ohjelmiston kehittäjien oletusten tueksi, tai sidosryhmien mielipiteitä ja uusia vaatimuksia (Khan et al. 2014). Yousuf et al. (2015) mukaan kyselystä on hyötyä, kun tarvitsee suurelta määrältä ihmisiä samaa tietoa. Kyselyn tuottama tieto on myös nopeasti analysoitavissa (Batra & Bhatnagar 2019).

Kysely voi aiheuttaa haasteita esimerkiksi vääristyneiden vastausten ja vähäisen vastausprosentin vuoksi. Vastausten vääristyneisyyttä voi aiheuttaa vastaajien miellyttämisenhalu. Tähän auttaa kyselyiden anonymiteetin varmistaminen. Anonyymiys on toisaalta usein myös kyselyiden vahvuus, koska sen avulla on mahdollista kysyä henkilökohtaisempiakin kysymyksiä. (Baxter & Courage 2005) Kyselyiden toisena haasteena on niiden monitulkintaisuus (Sharma & Pandey 2014; Yousuf & Asger 2015). Kysymyksen voi ymmärtää väärin (Yousuf & Asger 2015), eikä väärinymmärryksiä voi korjata samoin kuin esimerkiksi haastattelussa tai työpajassa (Sharma & Pandey 2014). Kyselyn avulla ei ole mahdollista saada kovin laajaa kuvaa kehitettävästä ohjelmistosta (Batra & Bhatnagar 2019).

Kyselyt sopivat parhaiten keskikokoisiin ja monimutkaisiin projekteihin (Gupta & Deraman 2019). Yousuf & Asger (2015) suosittelevat tekniikkaa kaupallisten ohjelmistojen vaatimusten kartoittamiseen. Tekniikan voi toteuttaa tarpeen mukaan nopeasti, ja kyselyyn vastaaminen on myös jokseenkin nopeaa. Kyselyä on suositeltava käyttää vaatimusten kartoittamisen keski- ja loppuvaiheessa. (Carriza et al. 2014; Gupta & Deraman 2019)

3.2.2 Yhteistekniikat

Prototyypin avulla kartoittaja voi varmistaa käyttäjiltä, että aiemmin kerätyt vaatimukset ovat oikeita (Anwar & Razali 2012). Käyttäjän palautteen perusteella aiempia vaatimuksia voidaan vahvistaa tai korjata, ja uusia vaatimuksia voidaan tunnistaa. Prototyyppi myös selventää sidosryhmille heidän vaatimustensa ja lopullisen toteutuksen välistä yhteyttä. (Khan et al. 2014) Pacheco et al. (2018) tunnistavat kirjallisuuskatsauksessaan prototyypin vahvuudeksi myös sen, että tekniikan avulla voi kerätä käyttäjärajapinnan vaatimuksia. Jos prototyyppi on kehityksellinen, sen käyttäminen vaatimusten kartoittamisessa voi myös nopeuttaa kehitysprosessia ja vähentää kustannuksia. (Khan et al. 2014)

Jos toteutettava ohjelmisto on monimutkainen, prototyypin toteuttaminen voi olla hidasta ja se voi vaatia paljon resursseja. Resurssien tarve voi myös kasvaa kesken toteutuksen, jos vaatimuksen muuttuvat. Käyttäjät voivat tottua tiettyyn prototyyppiin, jolloin voi ilmetä vastustusta muutoksille. (Yousuf & Asger 2015)

Guptan ja Deramanin (2019) mukaan prototyypit sopivat pienikokoisiin ja monimutkaisiin projekteihin. Prototyypit vaativat paljon aikaa toteuttamiseen (Gupta & Deraman 2019), eikä niitä suositella käytettävän, jos projektilla on tiukka aikataulu (Carrizo et al. 2014). Paperiprototyyppi on kuitenkin nopeampi toteuttaa (Rueda et al. 2020), joten se on todennäköisesti paras prototyyppivaihtoehto, jos aikataulu on tiukka. Prototyypin sopiva käyttö tapahtuu kartoittamisen keskivaiheilla (Gupta & Deraman 2019).

Pancheco et al. (2018) kirjallisuuskatsauksen mukaan prototyypillä voi kerätä funktionaalisia vaatimuksia. Rueda et al. (2020) käsittelevät tutkimuksessaan paperiprototyyppejä ja mainitsevat myös, että ne eivät sovellu ei-funktionaalisten vaatimusten kartoittamiseen. Batra ja Bhatnagar (2019) suosittelevat prototyypin käyttämistä uusien ohjelmistojen kehityksessä.

Käyttötapausten vahvuutena on se, että ne ovat helppoja ymmärtää, eivätkä vaadi käyttäjältä teknistä ymmärrystä. Tekniikan avulla saadaan loppukäyttäjän näkökulma mukaan vaatimusten kartoittamiseen. Toisaalta tekniikan luoma näkökulma on melko kapea, koska käyttötapauksissa käsitellään vain loppukäyttäjän näkemiä prosesseja. (Yousuf & Asger 2015) Batran ja Bhatnagarin (2019) mukaan käyttötapaukset ovat suhteellisen halpoja toteuttaa. Toisaalta Carrizo et al. (2014) suosittelee vain yhden käyttäjän osallistumista tilaisuuteen kerrallaan, joten tekniikan avulla saa tietoa vain pieneltä ihmisryhmältä. Käyttötapausten avulla voidaan kerätä myös hiljaista tietoa, eli vaatimuksia, joita käyttäjä ei osaa sanoittaa (Anwar et al. 2022). Canedo et al. (2023) tutkimuksen

mukaan käyttötapaus on yksi käytetyimmistä tekniikoista turvallisuusvaatimusten kartoittamisessa.

Haastattelun tavoin myös käyttötapaukset sopivat monipuolisesti erilaisiin projekteihin koosta ja monimutkaisuudesta riippumatta. Tekniikka vaatii toteutuksessa melko paljon aikaa, joten se ei sovi kovin tiukkoihin aikatauluihin. (Gupta & Deraman 2019) Tekniikka vaatii myös sidosryhmän edustajan paikan päälle (Batra & Bhatnagar 2019). Vaatimusten kartoittamisessa käyttötapauksia on sopiva käyttää prosessin keskivaiheilla (Gupta & Deraman 2019). Myös Yousuf ja Asger (2015) kertovat, että käyttötapauksia on sopiva käyttää sen jälkeen, kun ensimmäiset vaatimukset on kerätty, koska niiden luomiseen vaaditaan alustavaa tietoa ohjelmiston toiminnoista. Toisaalta Paetsch et al. (2003) mukaan käyttötapauksia voidaan käyttää ohjelmiston aikaisessa kehitysvaiheessa. Tämän perusteella käyttötapauksien sopiva ajankohta on vaatimusten kartoittamisen alku- ja keskivaihe.

3.2.3 Kognitiiviset tekniikat

Porrastustekniikka priorisoi vaatimuksia, ja vaatimuksia on helppo ymmärtää hierarkkisuuden vuoksi. Tekniikka mahdollistaa myös vaatimusten uudelleenkäytön, joka voi nopeuttaa kartoittamistilaisuutta ja vähentää resurssien tarvetta. (Yousuf & Asger 2015) Kun vaatimuksia tulee paljon, porrastus käy monimutkaisemmaksi. Vaatimuksia on vaikeaa muuttaa, poistaa tai lisätä rakenteessa. (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015; Batra & Bhatnagar 2019)

Porrastus sopii parhaiten käytettäväksi keskikokoisiin projekteihin (Gupta & Deraman 2019). Guptan ja Deramanin (2019) mukaan tekniikasta on eniten hyötyä kuitenkin monimutkaisissa projekteissa. Yousuf ja Asger (2015) eivät suosittele tekniikan käyttämistä uusien ohjelmistojen kehittämisessä. Porrastus toimii vaatimusten kartoittamisen alussa sekä keskivaiheilla, ja se voi toimia myös tiukassa aikataulussa (Carrizo et al. 2014; Gupta & Deraman 2019). Batra ja Bhatnagar (2019) eivät suosittele tekniikan käyttämistä silloin, kun vaatimuksia on paljon ja toteutettava ohjelmisto on monimutkainen.

Korttilajittelun avulla voidaan löytää ohjelmiston rakenteelle pohja, joka on toimiva käyttäjien mielestä. Tekniikka on helppo toteuttaa, ja toteutuksen voi tehdä myös verkossa, jolloin käyttäjän fyysistä läsnäoloa ei vaadita. (Yousuf & Asger 2015) Sidosryhmän edustajalta vaaditaan kuitenkin teknistä ymmärrystä ohjelmistosta korttilajitteluun osallistumiseksi (Batra & Bhatnagar 2019). Korttilajittelun avulla saadaan vain pinnallisia vaatimuksia, eikä tekniikka ota kantaa syvempiin prosesseihin. Tekniikan tulokset voivat olla myös vaihtelevia, jolloin yhdenmukaisia vaatimuksia ei saada. Tekniikan pinnallisuuden vuoksi syventävää selitystä vaatimuksille ei voi saada. (Yousuf & Asger 2015)

Korttilajittelu sopii parhaiten keskikokoisiin ja monimutkaisiin projekteihin (Gupta & Deraman 2019). Yousuf ja Asger (2015) eivät kuitenkaan suosittele tekniikan käyttöä, jos kehitettävä ohjelmisto on hyvin monimutkainen, iso ja hajanainen. Korttilajittelu toimii vaatimusten kartoittamisen alussa sekä keskivaiheilla, ja se voi toimia myös tiukassa aikataulussa (Gupta & Deraman 2019). Myös Yousufin ja Asgerin (2015) mukaan korttilajittelu on nopea toteuttaa.

Yousufin ja Asgerin (2015) mukaan repertory grid -tekniikan avulla voidaan eritellä elementtien samankaltaisuuksia ja eroavaisuuksia. Tekniikan avulla voidaan ehkäistä kartoittajan ennakkoluulojen vaikutusta vaatimuksiin ja saada käyttäjän näkemys paremmin huomioon. (Yousuf & Asger 2015) Repertory grid mahdollistaa jäljitettävyyden vaatimuksille, ja vaatimusten välisten yhteyksien löytämisen (Batra & Bhatnagar 2019). Porrastustekniikan tapaan myös repertory grid -tekniikan toteuttaminen vaatii kartoittajalta paljon osaamista kartoittamisesta ja ongelma-alueesta, sekä ymmärrystä tekniikasta (Carrizo et al. 2014). Repertory gridin tuottama informaatio on yksinkertaista perustietoa (Carrizo et al. 2014; Gupta & Deraman 2019), eli tekniikalla ei voi kerätä monimutkaisempia vaatimuksia (Batra & Bhatnagar 2019).

Repertory grid sopii parhaiten pieniin ja yksinkertaisiin projekteihin (Gupta & Deraman 2019). Repertory grid toimii vaatimusten kartoittamisen keskivaiheessa, ja se voi toimia myös tiukassa aikataulussa (Carrizo et al. 2014; Gupta & Deraman 2019).

3.2.4 Havaintotekniikat

Tarkkailussa ohjelmiston käyttäjällä ei tarvitse olla erityistä teknistä osaamista ohjelmistoon liittyen, käyttäjältä ei vaadita suurta kiinnostusta vaatimusten kartoittamiseen eikä käyttäjän tarvitse osata sanoittaa vaatimuksia (Carrizo et al. 2014). Kartoitetut vaatimukset ovat autenttisia ja luotettavia, koska kartoittaja on aidossa tilanteessa paikalla (Yousuf & Asger 2015; Batra & Bhatnagar 2019). Tarkkailun avulla voidaan vahvistaa aiemmin kartoitettuja vaatimuksia. Tekniikan avulla voidaan kerätä myös vaatimuksia, joita ohjelmiston käyttäjä ei osaa sanoittaa. (Yousuf & Asger 2015)

Kartoittajan on osattava spesifioida ja analysoida vaatimuksia tarkkailun perusteella (Sharma & Pandey 2014). Tarkkailu on myös kallis tekniikka, ja tilaisuuteen voi liittyä myös ylimääräisiä kustannuksia, kuten matkakulut (Batra & Bhatnagar 2019). Tarkkailu sopii parhaiten pieniin ja keskikokoisiin projekteihin, mutta siitä on eniten hyötyä monimutkaisemmissa projekteissa (Gupta & Deraman 2019). Tarkkailu kannattaa toteuttaa vaatimusten kartoittamisen alussa, ja tekniikka vaatii runsaasti aikaa toteuttamiseen (Carrizo et al. 2014; Gupta & Deraman 2019).

3.2.5 Yhteenveto

Taulukkoon 4 on kerätty luvun 3 keskeisimmät tulokset. Tuloksista on tiivistetty mitä varten ja missä tilanteissa tekniikoita on sopivaa käyttää sekä suurimmat rajoitteet.

Taulukko 4. *Työn keskeisimmät tulokset*

Tekniikka	Käyttö	Rajoitteet
Keskustelevat		
Haastattelut	◦kokonaiskäsitys ohjelmistosta ◦yksityiskohtaiset vaatimukset (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015)	◦tietoa vain pieneltä joukolta ◦vaatii toimialan ymmärrystä kartoittajalta (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015)
Työpajat	◦suurten ongelmien ratkaisu ◦kokonaiset vaatimukset (Shams-Ul-Arif et al. 2009) ◦sidosryhmien konfliktien ratkaisu (Yousuf & Asger 2015)	◦kaikkien osallistujien aikataulujen sovittaminen haasteena (Yousuf & Asger 2015)
Aivoriihet	◦kartoittamisen alkuvaihe (Gupta & Deraman 2019) ◦pääratkaisut ◦innovatiiviset ideat (Shams-Ul-Arif et al. 2009) ◦ei vaadi alan osaamista osallistujilta (Yousuf & Asger 2015) ◦suuret projektit	◦ei suurten ongelmien ratkaisuun (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015) ◦vaatii kokeneen kartoittajan (Carrizo et al. 2014)
Kyselyt	◦tieto laajalta ja maantieteellisesti hajanaiselta joukolta ◦ei vaadi alan osaamista vastaajalta (Carrizo et al. 2014) ◦kaupalliset tuotteet (Yousuf & Asger 2015)	◦vaatii huolellisen valmistelun ◦kysymysten monitulkintaisuus (Sharma & Pandey 2014; Yousuf & Asger 2015) ◦ei kokonaiskuvaa (Batra & Bhatnagar 2019)
Yhteistekniikat		
Prototyypit	◦vaatimusten vahvistaminen (Anwar & Razali 2012) ◦käyttäjärajapinnan vaatimukset ◦funktionaaliset vaatimukset (Pacheco et al. 2018) ◦pienet projektit (Gupta & Deraman 2019) ◦uudet ohjelmistot (Batra & Bhatnagar 2019)	◦työläs ja hidas toteuttaa (Gupta & Deraman 2019) ◦muutosvastarinta käyttäjien puolesta (Yousuf & Asger 2015)
Käyttötapaukset	◦ei vaadi alan osaamista osallistujalta ◦käyttäjän näkemät prosessit (Yousuf & Asger 2015) ◦hiljainen tieto	◦hidas toteuttaa (Gupta & Deraman 2019)

	(Anwar et al. 2022) ◦käyttö ensimmäisten vaatimusten jälkeen (Yousuf & Asger 2015; Gupta & Deraman 2019) ◦turvallisuusvaatimukset (Canedo et al. 2023)	
Kognitiiviset		
Porrastus	◦vaatimusten priorisointi ◦vaatimusten uudelleenkäyttö (Yousuf & Asger 2015) ◦keskikokoiset projektit (Gupta & Deraman 2019) ◦kartoittamisen alku- ja keskivaihe ◦tiukka aikataulu Carrizo et al. 2014; Gupta & Deraman 2019)	◦ei sovi uusien ohjelmistojen kehitykseen (Yousuf & Asger 2015) ◦ei sovi käytettäväksi, kun paljon vaatimuksia ◦vaatimusten muuttaminen jälkikäteen vaikeaa (Shams-Ul-Arif et al. 2009; Yousuf & Asger 2015; Batra & Bhatnagar 2019)
Korttilajittelu	◦ohjelmiston rakenne ◦ei vaadi käyttäjän fyysistä läsnäoloa (Yousuf & Asger 2015) ◦tiukka aikataulu (Yousuf & Asger 2015; Gupta & Deraman 2019) ◦yksinkertainen ohjelmisto (Yousuf & Asger 2015)	◦pinnalliset vaatimukset (Yousuf & Asger 2015) ◦vaatii osallistujalta teknistä osaamista ohjelmistosta (Batra & Bhatnagar 2019)
Repertory grid	◦ohjelmiston elementtien erittely ◦kartoittajan ennakkoluulojen poisto (Yousuf & Asger 2015) ◦vaatimusten väliset yhteydet (Batra & Bhatnagar 2019) ◦pienet ja yksinkertaiset projektit (Gupta & Deraman 2019)	◦tuottaa yksinkertaista tietoa (Carrizo et al. 2014; Gupta & Deraman 2019) ◦vaatii kartoittajalta paljon kokemusta (Carrizo et al. 2014)
Havaintotekniikat		
Tarkkailu	◦kun osallistuja ei osaa sanoittaa vaatimuksia ◦vaatimusten vahvistaminen (Yousuf & Asger 2015) ◦ei vaadi alan osaamista osallistujalta (Carrizo et al. 2014) ◦vaatimusten kartoittamisen aikainen vaihe (Carrizo et al. 2014; Gupta & Deraman 2019) ◦pienet ja keskikokoiset projektit (Gupta & Deraman 2019)	◦hidas toteuttaa (Carrizo et al. 2014; Gupta & Deraman 2019)

4. PÄÄTELMÄT

Tutkimuksen tavoitteena oli tutkia ohjelmistoprojektin kartoittamistekniikoita ja niiden käyttöä. Kartoittamisen ja vaatimustenkäsittelyn huolellinen toteuttaminen on erityisen tärkeää ohjelmistoprojektin onnistumisen kannalta. Tutkimuksessa vastattiin seuraaviin tutkimuskysymyksiin:

(1) Mitä tekniikoita vaatimusten kartoittamisessa käytetään?

(2) Minkälaisiin käyttötarkoituksiin eri kartoittamistekniikat sopivat?

Luvussa 2 kuvattiin mitä käytettävissä olevia tekniikoita on olemassa ja miten ne toimivat. Käsittelyyn otettiin 10 yleisesti käytössä olevaa tekniikkaa. Luvussa 3 tutkittiin tekniikoiden ominaisuuksia ja tekniikoille suositeltuja käyttökohteita. Tutkimus oli tekniikkalähtöinen, eli jokaista tekniikkaa käsiteltiin omissa kappaleissaan. Luvun 3 perusteella luotiin ohjenuora, jonka avulla voi vertailla eri tekniikoiden sopivuutta eri tilanteisiin.

Tekniikoiden suositeltavasta käytöstä voidaan löytää yhteisiä tilanteita osalle tekniikoista. Kartoittamisen alkuvaiheessa on suositeltavaa käyttää aivoriihtä, porrastusta ja tarkkailua. Kartoittamisen loppuvaiheeseen ei ole selkeää omaa tekniikkaa, mutta ainakin kyselyä on sopiva käyttää myös lopussa. Nopeaan aikatauluun suositellaan kyselyä, porrastustekniikkaa ja korttilajittelua. Erityisen hitaita toteuttaa ovat taas prototyypit, käyttötapaukset ja tarkkailu. Aivoriihet, kyselyt, käyttötapaukset ja tarkkailu ei vaadi sidosryhmän edustajalta alan osaamista, kun taas esimerkiksi korttilajittelu vaatii teknistä osaamista. Erityisesti haastattelut, aivoriihet ja repertory grid vaativat myös kartoittajalta paljon kokemusta. Haastattelut, työpajat ja aivoriihet tuottavat laajaa ja korkeatasoista tietoa, kun taas muut tekniikat tuottavat spesifimpää tietoa eri tarpeisiin, minkä perusteella nämä tekniikat voisivat olla hyvin yleiskäyttöisiä useimpiin projekteihin.

Tulokset antavat yleiskuvan eri kartoittamistekniikoiden käytöstä, mutta valittaessa tekniikkaa vaatimusten kartoittamiseen on tunnettava tekniikoiden ominaisuuksia ja rajoitteita tarkemmin. Jatkotutkimuksena olisi hyvä tehdä jokaisesta tekniikasta erikseen tarkempi selvitys. Tarkasteluun olisi hyvä ottaa myös laajemmin eri tekniikoita, koska esimerkiksi, kuten luvussa 2.2 on mainittu, Aflen ja Vasquez (2021) tunnistivat kirjallisuuskatsauksessaan 40 eri tekniikkaa. Mahdollisia tekniikoita on siis moninkertaisesti tässä tutkimuksessa käsiteltyihin 10:n tekniikkaan nähden. Myös uusia tekniikoita kehitetään vaatimusten kartoittamisen haasteiden taltuttamiseksi, esimerkiksi pelillistämisen tavoit-

teena on vähentää vaatimusten monitulkintaisuutta (Dar et al. 2022). Myös globaalit ilmiöt, kuten koronaviruspandemia (Ul Amin et al. 2021) ovat vaikuttaneet erilaisten tekniikoiden valintaan.

LÄHTEET

Alflen, N.C., Vasques Prado, E.P. (2021). Requirements elicitation techniques for software development: a systematic review of literature. AtoZ 10, 39. https://doi.org/10.5380/atoz.v10i1.77393

Ali Ramdhani, M., Sa'adillah Maylawati, D., Syakur Amin, A., Aulawi, H. (2018). Requirements Elicitation in Software Engineering. IJET 7, 772. https://doi.org/10.14419/ijet.v7i2.29.14254

Aljuhani, A. (2021). Multi-Criteria Decision-Making Approach for Selection of Requirements Elicitation Techniques based on the Best-Worst Method. IJACSA 12. https://doi.org/10.14569/IJACSA.2021.0121183

Anwar, F., Razali, R. (2012). A Practical Guide to Requirements Elicitation Techniques Selection - An Empirical Study. Middle-East Journal of Scientific Research 11.

Anwar, H., Khan, S.U.R., Iqbal, J., Akhunzada, A. (2022). A Tacit-Knowledge-Based Requirements Elicitation Model Supporting COVID-19 Context. IEEE Access 10, 24481–24508. https://doi.org/10.1109/ACCESS.2022.3153678

Batra, M., Bhatnagar, A. (2019). Requirements Elicitation Technique Selection: A Five Factors Approach. International Journal of Engineering and Advanced Technology (IJEAT) 8.

Baxter, K., Courage, C. (2005). Understanding Your Users: A Practical Guide to User Requirements Methods, Tools, and Techniques. Gulf Professional Publishing.

Belani, H., Pripuzic, K., Kobas, K. (2005). Implementing web-surveys for software requirements elicitation, in: Proceedings of the 8th International Conference on Telecommunications, 2005. ConTEL 2005. Presented at the 2005 Proceedings of the 8th International Conference on Telecommunications, IEEE, Zagreb, Croatia, 465–469. https://doi.org/10.1109/CONTEL.2005.185931

Bly, S. (1997). Field work: is it product work? interactions 4, 25–30. https://doi.org/10.1145/242388.242398

Bourque, P., Fairley, R.E. (2014). Guide to the Software Engineering Body of Knowledge, Version 3.0. IEEE Computer Society.

Canedo, E.D., Calazans, A.T.S., Silva, G.R.S., Costa, P.H.T., De Mesquita, R.P., Masson, E.T.S. (2022). Creativity and Design Thinking as Facilitators in Requirements Elicitation. Int. J. Soft. Eng. Knowl. Eng. 32, 1527–1558. https://doi.org/10.1142/S0218194022500607

Carrizo, D., Dieste, O., Juristo, N. (2014). Systematizing requirements elicitation technique selection. Information and Software Technology 56, 644–669. https://doi.org/10.1016/j.infsof.2014.01.009

Dar, H.S., Imtiaz, S., Lali, M.I. (2022). Reducing Requirements Ambiguity via Gamification: Comparison with Traditional Techniques. Computational Intelligence and Neuroscience 2022, 1–12. https://doi.org/10.1155/2022/3183411

Dey, S., Lee, S.-W. (2015). From requirements elicitation to variability analysis using repertory grid: A cognitive approach, in: 2015 IEEE 23rd International Requirements Engineering Conference (RE). Presented at the 2015 IEEE 23rd International Requirements Engineering Conference (RE), IEEE, Ottawa, ON, Canada, 46–55. https://doi.org/10.1109/RE.2015.7320407

Fernández, D.M., Wagner, S., Kalinowski, M., Felderer, M., Mafra, P., Vetrò, A., Conte, T., Christiansson, M.-T., Greer, D., Lassenius, C., Männistö, T., Nayabi, M., Oivo, M., Penzenstadler, B., Pfahl, D., Prikladnicki, R., Ruhe, G., Schekelmann, A., Sen, S., Spinola, R., Tuzcu, A., De La Vara, J.L., Wieringa, R. (2017). Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice. Empir Software Eng 22, 2298–2338. https://doi.org/10.1007/s10664-016-9451-7

Fransella, F., Bell, R., Bannister, D. (2004). A Manual for Repertory Grid Technique. 2. ed. Chichester : Wiley.

Gupta, A.K.G., Deraman, A. (2019). A framework for software requirement ambiguity avoidance. IJECE 9, 5436. https://doi.org/10.11591/ijece.v9i6.pp5436-5445

Haikala, I., Mikkonen, T. (2011). Ohjelmistotuotannon käytännöt, 12., uudistettu painos. ed. Helsinki : Talentum.

Hickey, A.M., Davis, A.M. (2004). A Unified Model of Requirements Elicitation. Journal of Management Information Systems 20, 65–84. https://doi.org/10.1080/07421222.2004.11045786

Hussain, A., Mkpojiogu, E.O.C., Kamal, F.M. (2016). The Role of Requirements in the Success or Failure of Software Projects 6.

Khan, S., Dulloo, A.B., Verma, M. (2014). Systematic Review of Requirement Elicitation Techniques. International Journal of Information and Computation Technology 4, 133–138.

Kotonya, G., Sommerville, I. (1998). Requirements engineering: processes and techniques, Worldwide series in computer science. J. Wiley, Chichester ; New York.

Kujala, S. (2008). Effective user involvement in product development by improving the analysis of user needs. Behaviour & Information Technology 27, 457–473. https://doi.org/10.1080/01449290601111051

Li, Jinyu, Ullah, A., Li, Jun, Nazir, S., Khan, H.U., Ur Rehman, H., Haq, A.U. (2020). Attributes-Based Decision Making for Selection of Requirement Elicitation Techniques Using the Analytic Network Process. Mathematical Problems in Engineering 2020, 1–13. https://doi.org/10.1155/2020/2156023

Mich, L., Sakhnini, V., Berry, D. (2023). To group or not to group? Group sizes for requirements elicitation. Information and Software Technology 160, 107229. https://doi.org/10.1016/j.infsof.2023.107229

Muqeem, M., Ahmad, S., Nazeer, J., Farooqui, Md.F., Alam, A. (2022). Selection of Requirement Elicitation Techniques: A Neural Network based Approach. IJACSA 13. https://doi.org/10.14569/IJACSA.2022.0130144

Okesola, O., Okokpujie, K., Goddy-Worlu, R., Ogunbanwo, A. (2019). Qualitative comparisons of elicitation techniques in requirement engineering. Journal of Engineering and Applied Sciences 14.
Pacheco, C., García, I., Reyes, M. (2018). Requirements elicitation techniques: a systematic literature review based on the maturity of the techniques. IET softw. 12, 365–378. https://doi.org/10.1049/iet-sen.2017.0144

Paetsch, F., Eberlein, A., Maurer, F. (2003). Requirements engineering and agile software development, in: WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. Presented at the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, IEEE Comput. Soc, Linz, Austria, 308–313. https://doi.org/10.1109/ENABL.2003.1231428

Palomares, C., Franch, X., Quer, C., Chatzipetrou, P., López, L., Gorschek, T. (2021). The state-of-practice in requirements elicitation: an extended interview study at 12 companies. Requirements Eng 26, 273–299. https://doi.org/10.1007/s00766-020-00345-x

Pohl, K. (2010). Requirements engineering: fundamentals, principles, and techniques. Springer, Heidelberg ; New York.

Pressman, R.S. (2005). Software Engineering, A Practitioner's Approach. McGraw-Hill.
Rueda, S., Panach, J.I., Distante, D., 2020. Requirements elicitation methods based on interviews in comparison: A family of experiments. Information and Software Technology 126, 106361. https://doi.org/10.1016/j.infsof.2020.106361

Rugg, G., McGeorge, P. (1997). The sorting techniques: a tutorial paper on card sorts, picture sorts and item sorts. Expert Systems 14, 80–93. https://doi.org/10.1111/1468-0394.00045

Schlosser, R. W., Wendt, O., Bhavnani, S., & Nail-Chiwetalu, B. (2006). Use of information-seeking strategies for developing systematic reviews and engaging in evidence-based practice: the application of traditional and comprehensive Pearl Growing. A review. International Journal of Language & Communication Disorders, *41*(5), 567-582.

Shams-Ul-Arif, M., Khan, M.Q., Gahyyur, S.A.K. (2009). REQUIREMENTS ENGINEERING PROCESSES, TOOLS/TECHNOLOGIES, & METHODOLOGIES. International Journal of Reviews in Computing 2, 41–56.

Sharma, S., Pandey, S.K. (2014). Requirements elicitation: Issues and challenges, in: 2014 International Conference on Computing for Sustainable Global Development (INDIACom). Presented at the 2014 International Conference on Computing for Sustainable Global Development (INDIACom), IEEE, New Delhi, India, 151–155. https://doi.org/10.1109/IndiaCom.2014.6828119

Ul Amin, T., Shahzad, B., Fazal-e-Amin, Shoaib, M. (2021). Economical Requirements Elicitation Techniques During COVID-19: A Systematic Literature Review. Computers, Materials & Continua 67, 2665–2680. https://doi.org/10.32604/cmc.2021.013263

Wagner, S., Fernández, D.M., Felderer, M., Vetrò, A., Kalinowski, M., Wieringa, R., Pfahl, D., Conte, T., Christiansson, M.-T., Greer, D., Lassenius, C., Männistö, T., Nayebi, M., Oivo, M., Penzenstadler, B., Prikladnicki, R., Ruhe, G., Schekelmann, A., Sen, S., Spínola, R., Tuzcu, A., Vara, J.L.D.L., Winkler, D. (2019). Status Quo in Requirements Engineering: A Theory and a Global Family of Surveys. ACM Trans. Softw. Eng. Methodol. 28, 1–48. https://doi.org/10.1145/3306607

Younas, M., Jawawi, D.N.A., Ghani, I., Kazmi, R. (2017). Non-Functional Requirements Elicitation Guideline for Agile Methods 9.

Yousuf, M., M.Asger, M.A. (2015). Comparison of Various Requirements Elicitation Techniques. IJCA 116, 8–15. https://doi.org/10.5120/20322-2408

LIITE A: HELMENKASVATUSARTIKKELIT