Käytettävyyden kehittämisen haasteet avoimen lähdekoodin ohjelmistoissa

Oulun yliopisto
Tietojenkäsittelytieteiden laitos
LuK-tutkielma
Aatos Lang
Pvm: 28.1.2016

Tiivistelmä

Avoimen lähdekoodin käyttö ohjelmistokehityksen työkaluna on kasvattanut yhä enenevissä määrin suosiostaan, mutta samalla ohjelmistot ovat saaneet osakseen kritiikkiä niiden heikommasta käytettävyydestä. Tämän tutkielman tavoitteena oli tunnistaa keskeisiä käytettävyyden kehittämisen haasteita, joita avoimen lähdekoodin ohjelmistojen kehitykseen liittyy. Lisäksi tarkoituksena oli löytää ohjelmistojen kehitysprosessista syitä, jotka vaikuttavat näiden haasteiden syntymiseen. Tutkielma toteutettiin perehtymällä aiheesta aiemmin tehtyihin julkaisuihin ja tutkimuksiin.

Tutkielman löydöksinä nousi esiin neljä keskeistä haastetta, joilla havaittiin olevan vaikutuksia avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen. Nämä neljä haastetta olivat: ohjelmistojen kehittäminen teknisesti osaavammille käyttäjille, käytettävyyden ammattilaisten puuttuminen, resurssien puute sekä kulttuuri ja asenteet.

Avainsanat
Avoimen lähdekoodin ohjelmisto, käytettävyys

Ohjaaja
Yliopisto-opettaja, Mikko Rajanen

Alkusanat

Motivaatio tämän työn kirjoittamiseen nousi omasta kiinnostuksestani käytettävyyden tutkimista kohtaan. Avoimen lähdekoodin ohjelmistot tarjosivat mielestäni mielenkiintoisen ja tietyllä tavalla uniikin ympäristön käytettävyyden tutkimisen kannalta, joten pohjasin työni aiheen nimenomaan tähän alueeseen. Haluan kiittää yliopisto-opettaja Mikko Rajasta tämän työn ohjaamisesta, ja rakentavasta palautteesta kirjoitusprosessin aikana.

Aatos Lang

Oulu, Tammikuu 28, 2016

Lyhenteet

HCI	Human-computer interaction

Sisällys

1. Johdanto

Tämän LuK-tutkielman tavoitteena oli tarkastella ja analysoida keskeisiä haasteita, joita avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen liittyy. Aihe oli ajankohtainen, sillä avoimen lähdekoodin käyttö ohjelmistokehityksen työkaluna on kasvattanut yhä enenevissä määrin suosiotaan (Nichols, Thomson & Yeates, 2001). Tänä päivänä käyttäjille on tarjolla lukuisia onnistuneita avoimen lähdekoodin ohjelmistoja (Nichols & Twidale, 2003), jotka pystyvät toiminnollisuudellaan haastamaan samankaltaisia, kaupallisten ohjelmistoyritysten tarjoamia omisteisia ohjelmistoja (Andreasen, Nielsen, Schrøder & Stage, 2006). Huolimatta siitä, että avoimen lähdekoodin ohjelmistot ovat yleistymässä ja niiden käyttäjämäärät ovat kasvussa (Nichols & Twidale, 2003), on niiden vaikutus tavalliselle käyttäjälle vielä pieni (Nichols et al., 2001). Ohjelmistot ovat usein suunniteltu teknisesti osaavimmille käyttäjille (Nichols et al., 2001; Nichols & Twidale, 2003), kuten ohjelmistokehittäjille (Andreasen et al., 2006), ja keskivertokäyttäjä valitseekin käyttöönsä usein vain omisteisia ohjelmistoja (Nichols & Twidale, 2003). Yhdeksi syyksi tähän Nichols ja Twidale (2003) näkevät avoimen lähdekoodin ohjelmistojen heikomman käytettävyyden, josta ohjelmistoja onkin kritisoitu (Andreasen et al., 2006).

Yleinen käsitys siitä, että avoimen lähdekoodinjärjestelmät omaavat huonon käyttöliittymän, on Bensonin, Muller-Proven ja Mzourekin (2004) mukaan ainakin osaksi perusteltua. Benson kumppaneineen (2004) on todennut, että avoimen lähdekoodin ohjelmistot ovat usein kehitetty kehittäjiltä kehittäjille ja palautesykli todellisten käyttäjien kanssa puuttuu, sillä ohjelmiston kehitysprosessissa on mukana vain harvoja käytettävyysammattilaisia.

Tutkiessaan avoimen lähdekoodin ohjelmistojen kehitysprosessia ja sen vaikutuksia heikkoon käytettävyyteen, Nichols ja Twidale (2003) tunnistivat prosessista muun muassa seuraavia piirteitä: ohjelmistojen kehittäjät eivät ole tyypillisiä loppukäyttäjiä, projekteissa ei ole mukana käytettävyyden ammattilaisia ja resurssien puutteen projekteissa. Benson kumppaneineen (2004) puolestaan totesivat GNOME-projektia tutkiessaan, että kehitysprosessista puuttuu selkeä käytettävyysmetodiikka. Monet kehittäjät osallistuvat projektiin vapaa-ajallaan, joten he ovat haluttomia seuraamaan raskaita kehitysprosesseja (Benson et al. 2004).

Pembertonin (2004) mukaan syy siihen, miksi avoimen lähdekoodin kehityksessä ei pystytä toteuttamaan käytettävyyttä, on ohjelmiston kehittäjien ja tavallisten käyttäjien välinen kuilu. Loppukäyttäjien löytämät käytettävyysongelmat eivät välttämättä ole ongelmia teknisesti osaavammille ohjelmiston kehittäjille, jolloin kehittäjät ovat haluttomia korjaamaan näitä ongelmia. Ohjelmointia osaamattomalla käyttäjällä ei taas ole osaamista korjata löydettyjä ongelmia. (Pemberton, 2004.)

Tämä tutkielma perustuu kirjallisuustutkimukseen, eli lähteenä on käytetty aiheesta aiemmin tehtyjä tutkimuksia ja julkaisuja. Varsinaista empiiristä tutkimusta ei siis ole tehty. Tutkielman rakenne on koostettu siten, että johdantoa seuraavassa luvussa on käyty läpi tutkielman kannalta keskeinen käsite käytettävyys. Luvussa on käyty läpi käytettävyyden määritelmää, sekä käytettävyyden asemaa avoimen lähdekoodin kehityksessä.

Luvussa kaksi on tehty katsaus avoimen lähdekoodin ohjelmistojen kehitykseen. Luvussa on ensin käyty läpi avoimen lähdekoodin ohjelmistojen kehitykseen liittyviä yleisiä piirteitä ja toimintatapoja. Tämän jälkeen on pyritty nostamana esiin keskeisiä eroja

perinteisten kaupallisten ohjelmistojen ja avoimen lähdekoodin ohjelmistojen kehityksen välillä.

Kolmannessa luvussa keskitytään itse varsinaiseen tutkimusongelmaan, eli avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämisen haasteisiin. Luvussa on pyritty vastamaan kahteen tälle tutkielmalle asetettuun tutkimuskysymykseen, jotka ovat:

1. Mitkä ovat keskeiset käytettävyyden kehittämisen haasteet avoimen lähdekoodin ohjelmistoissa?
2. Mikä tai mitkä tekijät kehitysprosessissa vaikuttavat näiden haasteiden esiintymiseen?

Luvussa on myös sivuttu hieman mahdollisia ratkaisuja näihin haasteisiin, mutta ne eivät ole pääasiallisena tutkimuskohteina tässä tutkielmassa. Luvut viisi ja kuusi pitävät sisällään vielä pohdinnan ja johtopäätökset sekä yhteenvedon.

2. Käytettävyys

Käytettävyydellä pyritään vastamaan kysymykseen siitä, kuinka hyvin käyttäjä pystyy käyttämään järjestelmän toimintoja, ja se koskettaa järjestelmän kaikkia niitä osa-alueita, joiden kanssa käyttäjä voi olla vuorovaikutuksessa (Nielsen, 1993). Tässä luvussa on ensin avattu käytettävyyden määritelmää, jonka jälkeen on pohdittu käytettävyyden asemaa avoimen lähdekoodin kehityksessä.

2.1 Käytettävyyden määritelmä

Käytettävyydelle on tarjolla useita erilaisia määritelmiä, jotka vaihtelevat lähteestä ja käyttökontekstista riippuen. Näistä yksi ja usein käytetty on Jakob Nielsenin (1993) esittelemä määritelmä, jonka mukaan käytettävyydellä voidaan perinteisesti tarkoittaa kokonaisuutta, joka koostuu seuraavista viidestä ominaisuudesta: opittavuudesta, tehokkuudesta, muistettavuudesta, virheiden määrästä ja tyydyttävyydestä. Se ei siis ole ohjelmiston yksittäinen tai yksiulotteinen ominaisuus, vaan useamman komponentin muodostama summa. (Nielsen, 1993.) Seuraavassa on avattu hieman edellä mainittujen viiden ominaisuuden merkityksiä.

- *Opittavuudella* tarkoitetaan sitä, että järjestelmän käytön tulisi olla helposti opittavissa, jotta käyttäjä voi nopeasti siirtyä työskentelemään sen pariin.
- *Tehokkuudella* tarkoitetaan sitä, että järjestelmän käytön tulee olla tehokasta, jotta opittua järjestelmän käytön, käyttäjällä on mahdollista työskennellä erittäin tuottavasti.
- *Muistettavuudella* tarkoitetaan sitä, että järjestelmän tulee olla helposti muistettavissa, jotta käyttäjä voi palata työskentelemään järjestelmän parissa pienen tauon jälkeenkin, ilman uudelleen opettelemista.
- *Virheiden määrä* järjestelmässä tulisi olla pieni, jolloin estetään käyttäjää törmäämästä moniin virheisiin käytön aikana. Mahdollisten virheiden sattuessa, niistä pitää olla myös helposti toivuttavissa.
- *Tyydyttävyydellä* tarkoitetaan sitä, että järjestelmän käytön tulee olla tyydyttävää, jolloin se miellyttää käyttäjiä. (Nielsen, 1993.)

Toinen yleisesti käytetty määritelmä on ISO (International Organization for Standardization) 9421-11 –standardin mukainen määritelmä käytettävyydelle. ISO 9241 pitää sisällään ergonomisia vaatimuksia käyttöpäätteiden kanssa tehtävään toimistotyöhön ja sen yhdennestätoista osasta löytyy määritelmä käytettävyydelle. Siinä käytettävyys on määritelty seuraavasti: ”Mitta, miten hyvin määrätyt käyttäjät voivat käyttää tuotetta määrätyssä käyttötilanteessa saavuttaakseen määritetyt tavoitteet tuloksellisesti, tehokkaasti ja miellyttävästi.” (International Organization for Standardization [ISO], 1998).

2.2 Avoimen lähdekoodin ohjelmistojen käytettävyys ja kritiikki

Käytettävyyden tutkiminen avoimen lähdekoodin ohjelmistoissa on noussut ajankohtaiseksi aiheeksi, johtuen ohjelmistojen leviämisestä (Zhao & Deek, 2005). Leviämisen myötä ohjelmistot ovat saaneet lisää teknisesti osaamattomampia käyttäjiä, ja avoimen lähdekoodin yhteisöt ovat alkaneet ymmärtää, etteivät vain he itse ole enää kehittämiensä ohjelmistojen käyttäjiä (Frishberg, Dirks, Benson, Nickell & Smith, 2002). Tämä on lisännyt tarvetta avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen, sillä kuten Nichols ja Twidalekin ovat todenneet (2003),

käytettävyysongelmat alkavat esiintyä juuri siinä vaiheessa, kun ohjelmistoja aletaan suunnittelemaan teknisesti osaamattomammille käyttäjille.

Avoimen lähdekoodin ohjelmistojen käytettävyyden tutkimusta on lisännyt myös se, että ihmisen ja tietokoneen välistä vuorovaikutusta tutkivat HCI (engl. human-computer interaction) -ammattilaiset ovat alkaneet kiinnostua aiheesta (Frishberg et al., 2002). Frishbergin ja kumppaneiden (2002) mielestä ohjelmistokehitystä tekevillä, ei-hierarkkisilla ja ilman maantieteellisiä rajoja toimivilla organisaatioilla, on paljon kerrottavaa HCI-yhteisölle. Vastaavasti Frishberg kumppaneineen (2002) näkevät myös, että HCI:tä harjoittavilla henkilöillä, kuten käytettävyysammattilaisilla ja vuorovaikutussuunnittelijoilla, on tarjottavaa avoimen lähdekoodin yhteisöille.

Yleinen käsitys ei tue sitä näkemystä, että avoimen lähdekoodin kehitys ja hyvä käytettävyys kulkisivat käsi kädessä. Tätä näkemystä on ollut taipumus perustella sillä, että käytettävyys on asia, jonka kehittäminen vaatii käytettävyyden ammattilaisten omistautunutta panostusta. Tämä argumentti pohjaa siihen oletukseen, että käytettävyyden ammattilaiset eivät osallistu avoimen lähdekoodin projekteihin. (Zhao & Deek, 2005.)

3. Avoimen lähdekoodin ohjelmistojen kehitys

Tämän tutkielman sisäistämisen kannalta on olennaista ymmärtää, miten ja millä tavoin avoimen lähdekoodin ohjelmistojen kehitystä toteutetaan. Tässä luvussa on pyritty luomaa selkeä yleiskuva avoimen lähdekoodin yhteisöjen keskeisistä toimintatavoista ja periaatteista. Luvussa on ensin käyty läpi avoimen lähdekoodin ohjelmistojen kehitykseen liittyviä yleispiirteitä ja toimintatapoja, jonka jälkeen on keskitytty eroihin kaupallisten ja avoimen lähdekoodin ohjelmistojen kehityksen välillä.

3.1 Yleispiirteet ja toimintatavat

Avoimen lähdekoodin projektit ovat tavallisesti ohjelmistokehittäjien muodostamia Internet-pohjaisia verkostoja tai yhteisöjä (Von Krogh & Von Hippel, 2003), joissa ympäri maailmaa levittäytyneet kehittäjät osallistuvat ja antavat panoksensa ohjelmistokehitykseen Internetin välityksellä (Hertel, Niedner & Herrmann, 2003). Olennaista avoimen lähdekoodin ohjelmistokehityksessä on se, että ohjelman lähdekoodi on julkisesti saatavilla, jolloin jokainen riittävän taitotason omaava henkilö voi osallistua projektiin milloin tahansa (Hertel et al., 2003). Projektiin voi osallistua esimerkiksi lataamalla ohjelmiston lähdekoodin ja tekemällä siihen korjauksia tai laajennuksia (Hertel et al., 2003).

Ohjelmistokehitys avoimen lähdekoodin projekteissa perustuu yleensä vapaaehtoisuuteen, eli kehittäjät tekevät työtä ilmaiseksi saamatta siitä palkkaa. Kehittämistä tehdään tavallisesti harrastepohjalta vapaa-ajalla tai tavallisen päivätyön ohessa. (Hertel et al., 2003.) Monet avoimen lähdekoodinohjelmistojen kehittämiseen osallistuvista henkilöistä tulevat niin sanotusta hakkerikulttuurista (Madey, Freeh, Tynan, 2002). Kyseinen kulttuuri perustuu satoihin tuhansiin ympäri maailmaa levittäytyneisiin ohjelmoijiin, jotka vapaaehtoisesti tuottavat ja jakavat ohjelmiaan ilman rahallista korvausta (Madey, Freeh, Tynan, 2002). Kehittäjien lukumäärät projekteissa vaihtelevat muutamista henkilöistä jopa moniin tuhansiin henkilöihin, ja kehitettävillä ohjelmistoilla voi olla jopa miljoonia käyttäjiä. Esimerkkejä suuren käyttäjämäärän omaavista avoimen lähdekoodin ohjelmistoista ovat muun muassa GNU/Linux käyttöjärjestelmä, palvelinohjelma Apache ja ohjelmointikieli Perl. (Von Krogh & Von Hippel, 2003.)

Virallisesti avoimella lähdekoodilla tarkoitetaan muutakin kuin, että ohjelmiston lähdekoodi on saatavilla. Täyttääkseen avoimen lähdekoodin määritelmän, lähteen täytyy olla uudelleenjaettavissa maksutta ja ilman rajoituksia. Sen täytyy noudattaa lisenssiä, joka mahdollistaa muutosten tekemisen alkuperäiseen työhön ja sallii sen käytön johdannaisten töiden tekemiseen. Alkuperäisestä ohjelmistosta tehtyjen johdannaisten töiden täytyy myös olla uudelleenjaettavissa samoilla ehdoilla kuin alkuperäisen työnkin. Lisenssejä, jotka täyttävät avoimen lähdekoodin määritelmän ovat muun muassa GNU Public License (GPL) ja Mozilla Public License. (O'Reilly, 1999.)

3.2 Eroja kaupallisten ohjelmistojen kehitykseen

Avoimen lähdekoodin ohjelmistojen kehitys poikkeaa useammalta osalta perinteisten kaupallisten ohjelmistojen kehityksestä. Siinä missä avoimen lähdekoodin kehitystä toteutetaan hajautetusti vapaaehtoisesti toimivien kehittäjien toimesta, tapahtuu kaupallinen ohjelmistokehitys yritysten keskitetyssä hallinnassa. Keskitetyn hallinnan avulla yritykset voivat sovittaa projektit ja aikataulut henkilöstön mukaan, kun taas vapaaehtoisesti työskentelevien kehittäjien kanssa tämä on haastavampaa. (Karels, 2003.)

Verrattuna avoimen lähdekoodin ohjelmistojen kehitykseen, noudattavat kaupalliset ohjelmistot lisensoinnin osalta vastakkaista lähestymistapaa. Ohjelmistoalalla toimivat yritykset hyödyntävät tekijänoikeuslakiin perustuvia lisensointijärjestelyitä, joiden avulla ne voivat tuottaa voittoa lisensointimaksujen muodossa. (Von Krogh & Von Hippel, 2003.) Kaupallinen ohjelmistokehitys onkin keskittynyt kehittämään tuotteita, jotka tuottavat voittoa kehittäjilleen, kun taas monien avoimen lähdekoodin ohjelmistojen lähtökohtana on jokin tietty ongelma, joka pyritään ratkaisemaan kehitettävän ohjelmiston avulla. (O'Reilly, 1999.). Olennainen asia, jossa avoimen lähdekoodin ohjelmistojen kehitys poikkeaa kaupallisien ohjelmistojen kehityksestä, on myös lähdekoodin avoimuus. Kaupallisten ohjelmistotuottajien keskittymien tuotteidensa myyntiin ja lisensointiin on johtanut siihen, että pääsy ohjelmistojen lähdekoodiin on tarkasti rajoitettu vain yrityksen sisäiseen käyttöön. (Von Krogh & Von Hippel, 2003.)

4. Käytettävyyden kehittämisen haasteet

Nicholsin ja Twidalen (2003) mukaan se, että avoimen lähdekoodin ohjelmistoissa on käytettävyysongelmia, ei itsessään ole merkittävää, sillä ongelmia löytyy kaikista interaktiivista ohjelmistoista. Tässä luvussa onkin keskitytty tutkimaan sitä, miten juuri avoimen lähdekoodin lähestymistapa vaikuttaa ohjelmistojen käytettävyyteen. Kappaleessa käydään läpi aiheesta aiemmin tehtyjen tutkimusten ja niiden löydösten perusteella tunnistettuja keskeisiä haasteita, joita avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen liittyy. Luvussa pyritään vastamaan johdantoluvussa esiteltyihin kahteen tutkimuskysymykseen, jotka olivat: 1) Mitkä ovat keskeiset käytettävyyden kehittämisen haasteet avoimen lähdekoodin ohjelmistoissa? ja 2) Mikä tai mitkä tekijät kehitysprosessissa vaikuttavat näiden haasteiden esiintymiseen?

4.1 Ohjelmistojen kehittämien teknisesti osaavammille käyttäjille

Raymond (2001) on todennut seuraavasti: ”Jokainen hyvä ohjelmistotyö alkaa raapimalla kehittäjän henkilökohtaista kutinaa”. Tämä pätee erityisesti avoimen lähdekoodin ohjelmistoihin, sillä useat niistä ovat saaneet alkunsa käyttäjän ryhtyessä ratkaisemaan jotain henkilökohtaista ongelmaa ohjelmiston avulla (O’Reilly, 1999). Zhaon ja Elbaumin (2002) mukaan lähes 60 % ohjelmistoista ovat saaneet alkunsa näin. Käytettävyyden kehittämisen kannalta tämä on haaste, sillä kun käyttäjä kehittää ohjelmistoa oman henkilökohtaisen ongelman ratkaisemiseksi, hänellä on taipumus tehdä ohjelmistoa itsellensä (Pemberton, 2004). Nicholsin ja Twidalen (2003) mukaan perinteiset käytettävyysongelmat alkavatkin nousta esiin siinä vaiheessa, kun ohjelmistoja aletaan suunnittelemaan teknisesti osaamattomammille käyttäjille.

Kirjoittaessaan käytettävyydestä ja sen kehittämisestä, yksi Nielsenin (1993) keskeisiä huomioita oli se, että ohjelmistojen suunnittelijat eivät ole niiden tyypillisiä käyttäjiä. Suunnittelijat eroavat loppukäyttäjistä usealla tapaa, kuten yleisen kokemuksen ja järjestelmätietämyksen suhteen. Syvemmän asiantuntemuksen järjestelmistä omaava kehittäjä voi pitää suunniteltua ikkunaa tai virheilmoitusta täysin järkevänä, kun taas loppukäyttäjä saattaa olla ymmärtämättä niitä laisinkaan. (Nielsen, 1993.) Avoimen lähdekoodin ohjelmistojen kohdalla tilanne on kuitenkin monesti se, että kehittäjät ovat itseasiassa ohjelmistojen tyypillisiä käyttäjiä. Tällöin vaikeastikin käytettävä ohjelmistot voivat olla täysin toimivia niiden ammattimaisemman kohdeyleisön kannalta. (Nichols & Twidale, 2003.) Ongelmana on nimeen omaan se, että ne eivät välttämättä palvele käytettävyydeltään teknisesti osaamattomampien käyttäjien tarpeita. Nicholsin Ja Twidalen (2003) mukaan avoimen lähdekoodin ohjelmistot epäonnistuvat loppukäyttäjän kannalta käytettävyydessä siinä, että käytettävyysongelmia katsotaan niin sanotusti väärin silmin, jonka vuoksi ne jäävät huomaamatta.

Tutkiessaan Greenstone nimisen avoimen lähdekoodin ohjelmiston käytettävyyttä, Nichols kumppaneineen (2001) löysivät monta sellaista esimerkkiä ohjelmiston käyttäytymisestä ja dokumentaatiosta, johon kehittäjät itse olivat tyytyväisiä, mutta jotka aiheuttivat merkittäviä ongelmia käyttäjille. Greenstonen kohdalla Nichols kumppaneineen (2001) olettivat, että osa näihin ongelmiin törmänneistä käyttäjistä olisivat luovuttaneen ja yrittäneen käyttää jotain vastaavaa ohjelmaa. Tyypillinen avoimen lähdekoodin käyttäjä tai kehittäjä, olisi puolestaan ainakin raportoinut ongelmasta ja ehkä ehdottanut siihen ratkaisua. (Nichols et al., 2001.)

Mikäli avoimen lähdekoodin ohjelmistot haluavat saavuttaa jalansijaa tavallisten käyttäjien keskuudessa, niin ohjelmistojen kehittäjien on Pembertonin (2004) mukaan

alettava ottamaan huomioon tavallisten käyttäjien tunnistamia käytettävyysongelmia. Koodausta osaamattomalle käyttäjälle ei juurikaan ole iloa siitä, että ongelmaa raportoitaessa hänelle kerrottaan esimerkiksi, missä ohjelmiston lähdekoodi on saatavilla, kun hänellä ei kuitenkaan ole tarvittavia taitoja muokata tai kehittää sitä. Tällöin käyttäjällä on ongelma, jonka ratkaisuun hän ei pysty itse vaikuttamaan. Mikäli tilanne tämän lisäksi on vielä se, että kehittäjät eivät puolestaan näe raportoitua ongelmaa relevanttina, jää se todennäköisesti ratkaisematta. Käytettävyyden kehittämiseksi kehittäjien on siis korjattava tunnistettuja ongelmia, vaikka ne eivät esiintyisivät ongelmina heille itsellensä tai teknisesti osaavammille käyttäjille. (Pemberton, 2004.)

4.2 Käytettävyyden ammattilaisten puuttuminen

Käytettävyyden ammattilaiset osallistuvat ohjelmistokehitykseen evaluoimalla järjestelmän käytettävyyttä erilaisten käytettävyyden tarkastusmenetelmien avulla. Heidän tehtävänään on löytää järjestelmästä mahdolliset käytettävyysongelmat ja määritellä sen käytettävyyden taso. (Botella, Alarcon & Peñalver, 2014.) Menetelmiä järjestelmän käytettävyyden evaluointiin on tarjolla useampia (Nielsen, 1993; Nielsen, 1994; Rosenbaum, 1989), joista toiset perustuvat pelkästään käytettävyysammattilaisten arviointeihin ja toiset taas pitävät sisällään myös käyttäjän osallistumisen (Rosenbaum, 1989).

Käytettävyysammattilaisten suorittamat asiantuntija-arvioinnit muodostuvat evaluointiprosessista, jossa yhdistyvät tuotteen kohdeyleisön ja erilaisten tehtävien tutkiminen, hyödyntäen samalla syvää tietämystä yleisistä käytettävyyssäännöistä (Rosenbaum, 1898). Yksi yleinen asiantuntija-arviointiin perustuva menetelmä on heuristinen evaluointi, jonka tarkoituksena on löytää mahdolliset käytettävyysongelmat käyttöliittymästä, jotta ne voidaan ottaa huomioon suunnitteluprosessissa (Nielsen, 1993). Siinä pieni joukko käytettävyysammattilaisia arvioi yksitellen käyttöliittymää, käyttäen apunaan tunnistettuja käytettävyysperiaatteita eli heuristiikoita (Hollingsed & Novick, 2007; Nielsen, 1993). Evaluoinnin perustella löydetyille käytettävyysongelmille määritellään vakavuus ja niiden sijainti käyttöliittymässä (Hollingsed & Novick, 2007). Empiirinen ja ehkä käytetyin menetelmä käytettävyyden evaluointiin, jossa mukaan otetaan myös käyttäjät, on käyttäjätestaus (Nielsen, 1994). Nielsenin (1993) mukaan käyttäjätestaus todellisten käyttäjien kanssa on käytettävyysmenetelmistä keskeisin, sillä sen avulla saadaan tietoa siitä, miten ihmiset käyttävät tietokoneita, ja minkälaisiin käytettävyysongelmiin he törmäävät testattavan järjestelmän kanssa.

Käytettävyysammattilaiset ja käytettävyystiimit toimivat siis kommunikoijana käyttäjien ja ohjelmistonkehittäjien välillä. Tämä tarkoittaa sitä, että heidän on tehtävä yhteistyötä molempien osapuolten kanssa. Käyttäjiltä on kerättävä tietoa heidän toimintatavoistaan, ja tämä tieto on puolestaan välitettävä ohjelmistokehittäjille. (Borghom & Madsen, 1999.) Mikäli käytettävyysammattilaiset otetaan mukaan ohjelmistokehitykseen jo alkuvaiheessa, osallistuvat he usein evaluoinnin lisäksi myös ohjelmiston suunnitteluprosessiin esimerkiksi prototypoinnin kautta (Borgholm & Madsen, 1999).

Avoimen lähdekoodin kehitykselle on tyypillistä se, että projekteissa on harvoin mukana käytettävyyden ammattilaisia (Benson et al., 2004; Nichols & Twidale, 2003). Ammattilaisten puuttuminen on merkittävä haaste käytettävyyden kehittämiselle, sillä kuten Zhao ja Deek (2005) ovat myös todenneet, käytettävyyden on argumentoitu olevan asia, jonka kehitys vaatii ammattilaisten omistautunutta panosta. Ammattilaisten puute vaikeuttaa myös kommunikaatiota todellisten loppukäyttäjien kanssa, jolloin heiltä ei saada kehityksen kannalta tärkeää palautetta (Benson et al., 2004). Kyseinen ongelma ei koske pelkästään perinteisiä, kehittäjän omiin tarkoituksiin kehitettäviä ohjelmistoja,

vaan myös isompia avoimen lähdekoodin projekteja (Nichols & Twidale, 2006). Esimerkiksi Trudellen (2002) listatessa Mozilla projektin parissa opittuja asioita, oli yksi opetus se, että varmistetaan käyttöliittymäsuunnittelijoiden osallistuminen avoimen lähdekoodin kehitykseen.

Aikaisemmat tutkimukset (Benson et al., 2004; Nichols & Twidale, 2003; Nichols & Twidale 2006; Zhao & Deek, 2005) ovat nostaneen esille useita eri tekijöitä, jotka selittävät ammattimaisen käytettävyyden puuttumista avoimen lähdekoodin projekteissa. Monen projektin kohdalla yksi näistä tekijöistä on yksinkertaisesti resurssien puute. Siinä missä kaupalliset yritykset voivat tarvittaessa palkata käytettävyysammattilaisia ohjelmistokehitysprosesseihinsa, ei vapaaehtoisluonteisilla avoimen lähdekoodin yhteisöillä ole tätä mahdollisuutta. Tämä tarkoittaa sitä, että erilaiset käytettävyysaktiviteetit, kuten asianmukaiset käyttäjäevaluoinnit, eivät ole toteutettavissa. (Nichols & Twidale, 2006.)

Bensonin ja kumppaneiden (2004) mukaan avoimen lähdekoodin projektit eivät myöskään ole helpoin paikka toimia käytettävyysammattilaisena. Päinvastoin se voi olla erittäin haastavaa, johtuen projektien hajautetusta ja insinöörivetoisesta luonteesta, joka voi olla ristiriidassa käytettävyyden suunnittelumenetelmien kanssa. Avoimen lähdekoodin projektien hajautuneisuuden takia, käytettävyysammattilaisten on pystyttävä kommunikoimaan ympärimaailmaan levittäytyneiden kehittäjien kanssa, ja välitettävä käytettävyyttä yhteisöille, joilla ei välttämättä ole siitä hirveästi kokemusta. (Benson et al., 2004). Jopa se, mitä käytettävyydellä tarkoitetaan, voi olla osalle avoimen lähdekoodin kehittäjälle hämärän peitossa (Rajanen, Iivari & Anttila, 2011).

Joidenkin julkaisujen (Nichols et al., 2001; Nichols & Twdiale, 2003) mukaan käytettävyyden ammattilaisten osallistumattomuuteen voi vaikuttaa se, että avoimen lähdekoodin yhteisöt eivät näyttäydy ulkopuolisille toimijoille kovinkaan avoimena tai vastaanottavaisina. Niin sanottuun hakkerikulttuuriin perustuvat yhteisöt (Madey, Freeh, Tynan, 2002) voivat olla erittäin vastaanottavaisia toisille ympäri maailmaa levittäytyneille hakkeriluontoisille kehittäjille, kun taas ei niinkään vastaanottavaisia henkilöille, jotka tulevat hakkerikulttuurin ulkopuolelta (Nichols & Twidale, 2003).

Nichols ja Twidale (2003) ovat nostaneet esille myös joukon yksinkertaisempia syitä, jotka mahdollisesti vaikuttavat käytettävyyden ammattilaisten ja heidän mukanaan tuoman ammattimaisen käytettävyyden puutteeseen avoimen lähdekoodin projekteissa. Yksi näistä on muun muassa se, että käytettävyysammattilasten määrä suhteessa hakkereihin, on yksinkertaisesti niin paljon pienempi, jolloin heitä ei ole tarpeeksi käytettävissä. Käytettävyysammattilaiset eivät myöskään ole välttämättä samalla tavalla kiinnostuneita avoimen lähdekoodin lähestymistavasta kuin hakkeriluontoiset kehittäjät. Toisaalta käytettävyysammattilaisten puutetta voi selittää sekin, että avoimen lähdekoodinprojekteissa ei ole perinteisesti ollut tarvetta ammattimaiselle käytettävyydelle. (Nichols & Twidale, 2003.)

4.3 Resurssien puute

Kuten aiemmin jo todettiin, avoimen lähdekoodin kehitys perustuu yleensä vapaaehtoisuuteen, jonka vuoksi projektit joutuvat toimimaan usein hyvin pienillä budjeteilla ja resursseilla (Nichols & Twidale, 2003). Vapaaehtoisuus ja resurssien puute oli myös yksi syy siihen, että avoimen lähdekoodin yhteisöillä ei ole mahdollisuutta palkata kehitykseen käytettävyyden ammattilaisia, toisin kuin kaupallisilla yrityksillä (Nichols & Twidale 2003; Nichols & Twidale 2006). Resurssien puutetta selittää se, että projektien taustalla on harvoin minkäänlaista instituutiota tai kaupallista toimijaa, joka

maksaisi vapaaehtoisesti kehitykseen osallistuville kehittäjille palkkaa heidän tekemästään työstä (Hertel et al., 2003; Nichols & Twidale, 2006). Valoa tähän haasteeseen voi kuitenkin jatkossa tuoda se, että avoimen lähdekoodin ohjelmien kasvatettua merkittävyyttään, yritykset ovat alkaneet osallistumaan projekteihin tuoden samalla mukanaan merkittäviä resursseja yhteisöjen käyttöön (Nichols & Twidale, 2006).

Resurssien puute on nostettu monessa julkaisuissa (Andreasen et al., 2006; Nichols & Twdiale, 2003; Nichols & Twdiale 2006; Zhao & Deek, 2005) yhdeksi keskeiseksi haasteeksi käytettävyyden kehittämiselle. Sen lisäksi, että se vaikutta ulkopuolisten käytettävyysammattilaisten ja teknisten osaajien puutteeseen, asettaa se muitakin haasteita käytettävyyden kehittämiselle. Esimerkiksi tilojen, kuten asianmukaisten käytettävyyslaboratorioiden hankinta, ja laajamittaisten testien suorittaminen, ei resurssien puutteen takia ole mahdollista. Joissakin yhteisöissä onkin esimerkiksi alettu keskustelemaan ideasta, josko käytettävyyslaboratorioita voitaisiin pyytä lahjoituksena yhteisöjen käyttöön. Näin ollen käytettävyyttä voitaisiin tehostaa ja kehittää pienemmilläkin budjeteilla. (Nichols & Twidale, 2003.)

Bensonin ja kumppaneiden (2004) mukaan suurilla yrityksillä olisi paljon potentiaalia osallistua ja panostaa resurssejaan avoimen lähdekoodin projektien kehitykseen. Käytettävyysosaamisen tarjoamisella voisi olla positiivisia vaikutuksia sekä avoimen lähdekoodin yhteisöille että yrityksille itsellensä. Ennen kaikkia tämä yhteistyö voisi hyödyttää ohjelmistojen käyttäjiä, heidän saadessaan parempaa käytettävyyttä. Tapoja, joilla yritykset voivat osallistua ja tarjota resurssejaan avoimen lähdekoodin ohjelmistojen kehitykseen, on monia. Yksi hyvä esimerkki on Sun Microsystems, Inc (siirtyi Oraclen omistukseen vuonna 2010 (Oracle, 2010)), joka on osallistunut käytettävyyden kehittämiseen monissa avoimen lähdekoodin projekteissa, kuten Netbeans, GNOME ja OpenOffice.org (nykyisin Apache OpenOffice (Robweir, 2012)). Sun on osallistunut edellä mainittujen projektien kehitykseen muun muassa tarjoamalla yhteisöille monenlaista osaamista ja henkilöstä, kuten käytettävyysammattilaisia, käyttöliittymä- ja vuorovaikutusosaamista sekä kielellistä taitoa esimerkiksi kääntämistyöhön. Projekteihin osallistuneet käytettävyystiimit ovat puolestaan tarjonneet paljon tärkeää dokumentaatiota muun muassa käyttöliittymäsuunnittelua ja käyttäjien palautteita koskien. Tämän lisäksi käytettävyyden ammattilaiset ovat suorittaneet erinäisiä käytettävyystutkimuksia, jotka ovat raportoitu yhteisöille. (Benson et al., 2004.) Nicholas ja Twidale (2006) puolestaan kertovat, että yksi mahdollinen tapa kaupallisille yrityksille osallistua kehitykseen, on yksinkertaisesti asettaa omisteinen käyttöliittymä avoimen lähdekoodin kannan päälle, josta yhtenä esimerkkinä on Applen OS X.

Kaupallisten yritysten osallistuminen käytettävyyden kehittämiseen voi siis helpottaa merkittävästi puuttuvista resursseista aiheutuvia haasteita projekteissa. Nichols ja Twidale (2003) näkevät kuitenkin, että vaikka yritykset osallistuisivat avoimen lähdekoodin kehitykseen, eivät ne todennäköisesti ole valmiita panostamaan siihen niin paljon resursseja kuin suurten omisteisten ohjelmien kehittämiseen. Nicholas ja Twidale (2003) uskovat myös, että mikäli avoimen lähdekoodin kehityksen käytettävyysresursseissa ei näy kasvua, tai korvaavia menetelmiä käytettävyyden kehittämiselle ei löydetä, tulee käytettävyys olemaan jatkossakin rajoittautunutta.

4.4 Kulttuuri ja asenteet

Yhtenä haasteena avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiselle on noussut esille yhteisöjen kulttuuri ja kehittäjien asenteet (Nichols & Twidale, 2003; Nichols et al. 2001; Frhisberg et al., 2002; Iivari, Rajanen & Hedberg, 2014). Kuten jo aiemmin mainittiin, hakkerikulttuurista voimansa ammentavat avoimen lähdekoodin

yhteisöt, voivat osoittautua ulkopuolisille henkilöille hyvinkin suljetuiksi, ja muualta kuin kehittäjien keskuudesta tuleva panostus ohjelmistojen kehittämiseen on pientä. (Nichols & Twidale, 2003; Nichols et al., 2001.) Projekteihin vapaaehtoisesti osallistuvilla kehittäjillä on taipumus työskennellä heille itsellensä mielenkiintoisten aiheiden ja ominaisuuksien parissa, jolloin osamaattomammille käyttäjille tärkeiden ominaisuuksien kehittämien voi jäädä vähemmälle huomioille (Nichols & Twdiale, 2003; Nichols et al., 2001).

Avoimen lähdekoodin yhteisöjen suhtautumista ja vastaanottavaisuutta ulkopuoliseen käytettävyyteen voi osaltaan ohjailla mahdolliset stereotypiat. Siinä missä HCI-yhteisöt ovat luoneet avoimen lähdekoodin kehittäjistä tukun erilaisia stereotypioita, voi myös vastapuolella olla ennakkokäsityksiä HCI-ammattilaisista. Avoimen lähdekoodin kehittäjät nähdään usein nimenomaan hakkereina, jotka eivät juurikaan kunnioita perinteisiä auktoriteetteja tai muodollisia prosesseja. Avoimen lähdekoodin kehittäjät puolestaan voivat pitää HCI:n harjoittajia, kuten käyttöliittymäsuunnittelijoita ja käytettävyysasiantuntijoita tahoina, jotka eivät pysty tuottamaan kehityksen kannalta mitään hyödyllistä, kuten esimerkiksi koodia. Heidät saatetaan nähdä henkilöinä, joiden pääasiallisena tehtävänä on vain viilailla käyttöliittymäelementtien välisiä etäisyyksiä, ja suunnitella kaikki näyttämään samalta. (Frhisberg et al., 2002.)

Iivari kumppaneineen (2014) ovat tutkimuksessaan esittäneen empiirisiä tuloksia ulkoisen käytettävyyden vastaanottamisesta avoimen lähdekoodin projekteissa. Tutkimuksessa yliopisto-opiskelijoista muodostetut käytettävyystiimit osallistuivat avoimen lähdekoodin projekteihin, suunnittelemalla ja toteuttamalla erilaisia käytettävyysaktiviteetteja, kommunikoimalla projektiyhteisöjen kanssa ja keräämällä dataa käytettävyystyön vaikutuksista projekteihin. Tuloksista käy ilmi, että käytettävyyden vieminen ja vaikutusten aikaansaaminen, ei kaikkien tutkittujen projektien kohdalla onnistunut. Seitsemässä tutkitussa projektissa, kolmen projektin kohdalla, käytettävyystiimien toiminnalla ei ollut vaikutuksia ohjelmistokehitykseen. Niiden projektien kohdalla, jossa käytettävyysaktiviteeteilla onnistuttiin saamaan aikaan muutoksia, tulokset viittaavat siihen, että käytettävyystiimien on tärkeä osallistua kehitykseen heti sen alkuvaiheessa. Mikäli osallistuminen tapahtuu myöhemmässä vaiheessa kehitystä, viittaavat löydökset siihen, että käytettävyystiimien on nähtävä enemmän vaivaa käytettävyyden viemiseksi yhteisöön. Kyseisessä tutkimuksessa tämä tarkoitti ymmärryksen hankkimista tuotteesta, käytettävyysmotivaation etsimistä ja keskeisten päätöksentekijöiden tavoittamista projekteissa. (Iivari et al., 2014.)

Käytettävyyden ja käytettävyyskulttuurin liittämiseksi osaksi avoimen lähdekoodin kehitystä, voi sen onnistumisen mahdollisuutta lisätä oikeanlaisen lähestymistavan valinta (Rajanen et al., 2011; Rajanen, Iivari & Anttila, 2012). Tutkiessaan erilaisia lähestymistapoja käytettävyyden viemiseksi avoimen lähdekoodin kehitykseen, Rajanen kumppaneineen (2011) huomasivat, että perinteisessä ohjelmistokehityksessä laajalti hyödynnetty konsultatiivinen lähestymistapa, ei välttämättä ole toimivin vaihtoehto avoimen lähdekoodin kontekstiin. Sen sijaan, että käytettävyysammattilaiset toimisivat ulkopuolisina konsultteina, kehottaa Rajasen ja kumppaneiden (2011) löydökset osallistuvampaan lähestymistapaan. Tutkimuksessaan Rajanen kumppaneineen (2011) kävivät läpi neljä erilaista avoimen lähdekoodin projektia, joihin yliopisto-opiskelijoista muodostetut käytettävyystiimit tarjosivat käytettävyysosaamistaan. Lähestymistavan vaikutuksia käytettävyystoimien onnistumiseen tutkittiin siten, että kahdessa projektissa käytettävyystiimit osallistuivat kehitykseen hyödyntäen perinteistä konsultatiivista lähestymistapaa, ja toisissa kahdessa projektissa osallistuvaa lähestymistapaa. Tutkimuksen löydökset osoittavat, että konsultatiivisella lähestymistavalla ei onnistuttu saamaan aikaan käytettävyysvaikutuksia. Osallistuvalla lähestymistavalla, jossa

käytettävyystiimit pyrkivät ennen käytettävyysaktiviteettien suorittamista hankkimaan näkyvyyttä ja arvostusta yhteisön jäsenten keskuudessa, nähtiin sen sijaan olevan enemmän vaikutusta. Sen avulla käytettävyystiimit onnistuivat nostamaan esille käytettävyyden merkitystä kehityksessä, ja saamaan ohjelmiston ydinkehittäjät kiinnostumaan asiasta. Osallistuvassa lähestymistavassakin nousi esiin kuitenkin omat haasteensa. Etenkin kolmannen projektin kohdalla, käytettävyystiimiin osoittautui hyvin hankalaksi hankkia tarpeeksi näkyvyyttä yhteisössä, jonka vuoksi kehityksessä ei onnistuttu saamaan aikaan käytettävyysvaikutuksia. (Rajanen et al., 2011.)

Lisää empiirisiä tuloksia osallistuvan lähestymistavan vaikutuksista on tarjonnut Rajasen ja kumppaneiden (2012) tutkimus, jossa myöskin tarkasteltiin käytettävyysaktiviteettien viemistä avoimen lähdekoodin kehitykseen. Tutkimuskohteeksi kyseisessä tutkimuksessa oli valittu avoimen lähdekoodin peliprojekti, jonka kehitykseen opiskelijoista muodostetut käytettävyystiimit osallistuivat muun muassa suunnittelemalla ja toteuttamalla erilaisia käytettävyysaktiviteetteja, tarjoamalla koodausosaamista ja kommunikoimalla projektiyhteisön kanssa käytettävyysaktiviteettien löydösten pohjalta. Etenkin yhden käytettävyystiimin kohdalla pyrittiin keräämään tietoa juuri osallistuvan lähestymistavan vaikutuksista ohjelmiston käytettävyyteen. Tutkimuksen tulokset osoittivat selkeästi, että osallistuvalla lähestymistavalla onnistuttiin saamaan aikaan muutoksia ohjelmiston käyttöliittymään, ja samalla parantamaan sen käytettävyyttä. Yksi selkeä merkki käytettävyysaktiviteettien vaikutuksesta kehitykseen oli se, että niiden pohjalta tehtiin muutoksia ohjelmiston lähdekoodiin. Rajasen ja kumppaneiden (2012) tutkimus nosti selkeästi esiin sen, että käytettävyysvaikutusten aikaan saamiseksi, käytettävyystiimin on tultava tunnistetuksi osaksi kehittäjäyhteisöä ja hankittava arvostusta heidän keskuudessaan. (Rajanen et al., 2012.)

Siitä huolimatta, että tunnettavuuden ja arvostuksen hankkiminen yhteisön keskuudessa voi edesauttaa käytettävyyden viemistä avoimen lähdekoodin kehitykseen, voi se myös osaltaan asettaa omat haasteensa (Rajanen & Iivari, 2013). Rajanen ja Iivari (2013) ovat tutkimuksessaan tarkastelleet käytettävyysammattilaisten hyödyntämien HCI-filosofioiden ja avoimen lähdekoodin filosofioiden yhteensovittamista. HCI:n keskeisiä periaatteita on se, että käytettävyysammattilaiset toimivat kommunikoijina ohjelmiston kehittäjien ja loppukäyttäjien välillä. Rajasen ja Iivarin (2013) löydökset osoittavat kuitenkin, että mikäli käytettävyysammattilaiset ajautuvat arvostusta hankkiessaan panostamaan voivaransa liiaksi ohjelmiston tekniseen kehitykseen ja koodin tuottamiseen, voi heidän alkuperäinen tehtävänsä jäädä osaltaan suorittamatta. Tällöin käytettävyysammattilainen voi osaltaan luopua HCI-filosofiastaan, ja keskittyä käytettävyyden sijasta enemmän ohjelmiston kehittämiseen. (Rajanen & Iivari, 2013.)

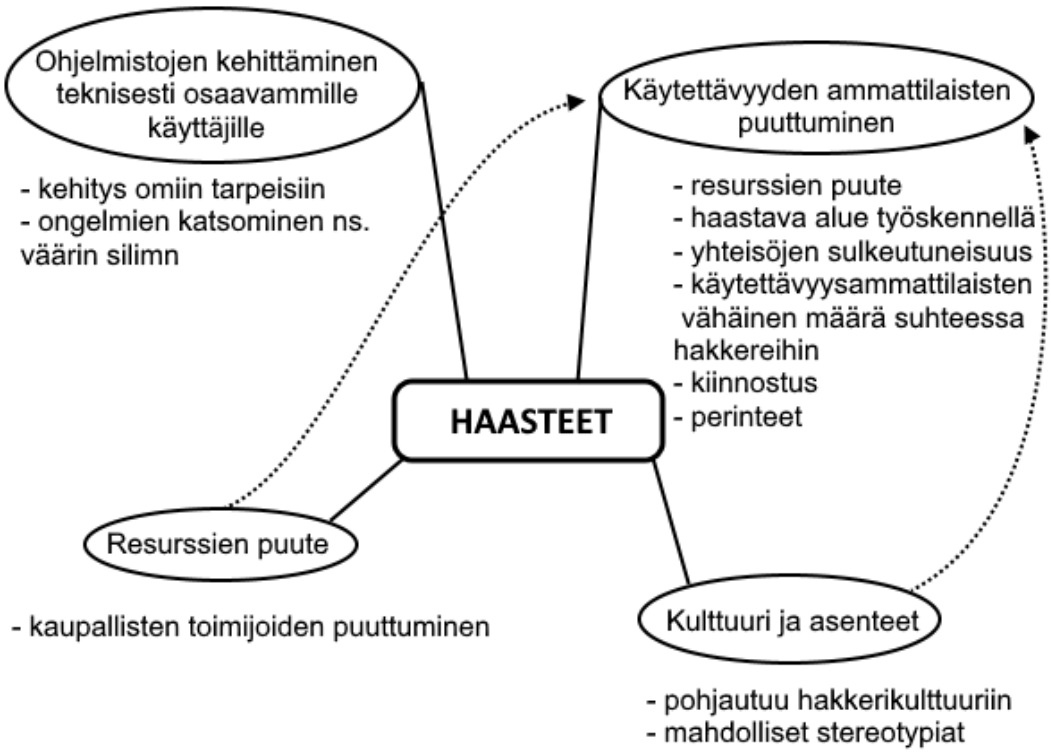
Sen lisäksi, että kulttuurit ja asenteet voivat vaikuttaa yhteisön ulkopuolelta tulevan käytettävyystyön vastaanottamiseen, voivat ne myös ohjailla toimintaa yhteisön sisällä (Nichols & Twdiale 2003, Nichols et al., 2001). Nichols ja Twidale (2003) näkevät, että kannustimet avoimen lähdekoodin kehityksessä, ajavat kehittäjiä mieluummin työskentelemään ohjelmistojen toiminnallisuuden, kuin käytettävyyden parissa. Yhtenä kannustimena monelle kehittäjille toimii kunnioituksen hankkiminen yhteisön keskuudessa. Tällaista kunnioitusta yhteisön keskuudessa voidaan hakea ratkaisemalla jokin vaikea ongelma tai haaste, joka tarkoittaa yleensä jonkin toiminnallisuuden lisäämistä tai korjaamista. (Nicholas & Twdiale, 2003.) Nichols kumppaneineen (2001) näkevät, että vaikeiksi ajateltujen algoritmisten ongelmien ratkaisemisella, voi ansaita yhteisössä enemmän mainetta kuin käytettävyys ongelmien ratkaisemisella. Tätä tukee myös Nicholsin & Twidalen (2003) näkemys, jonka mukaan toiminnallisuuden lisäämisen tai koodin optimointi, ovat kehittäjälle mahdollisuuksia näyttää osaamista muille yhteisön henkilöille. Tällainen kulttuuri, jossa käytettävyysongelmat nähdään

vähempiarvoisena, voi siis helposti johtaa siihen, että kehittäjillä ei ole motivaatiota paneutua niihin (Nichols & Twidale, 2003).

5. Pohdinta ja johtopäätökset

Tämän tutkielman tarkoituksena oli analysoida ja tarkastella keskeisiä käytettävyyden kehittämisen haasteita avoimen lähdekoodin ohjelmistoissa. Työssä käytiin läpi sitä, minkälaisia haasteita aiemmissa julkaisuissa ja tutkimuksissa on löydetty, ja perehdyttiin siihen, mitkä tekijät avoimen lähdekoodin kehityksessä vaikuttavat niiden esiintymiseen. Olemassa oleva lähdemateriaali tarjosi tutkielman kannalta relevanttia tietoa ja osoitti, että avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämistä on tutkittu. Aiemmat julkaisut olivat onnistuneet tunnistamaan useita käytettävyyden kehittämisen haasteita ja niiden taustalla olevia syitä, joten työlle asetettuihin tutkimuskysymyksiin saatiin vastauksia.

Kuvassa 1 on kuvattu käsitekarttamaisesti tutkielman löydöksiä. Kuvan ympyrät kuvaavat keskeisiä haasteita, joita avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen liittyy. Kunkin haasteen alla on esitetty tekijöitä, jotka vaikuttavat kyseisten haasteiden esiintymiseen. Katkoviivoilla esitetyt nuolet kuvaavat puolestaan haasteiden välisiä vaikutussuhteita.



Kuva 1. Tutkielman keskeiset löydökset

Kuten kuvasta 1 huomataan, tämän tutkielman tuloksena onnistuttiin löytämään neljä keskeistä haastetta, joita avoimen lähdekoodin käytettävyyden kehittämiseen liittyy. Lisäksi näiden haasteiden taustalla vaikuttavia syitä onnistuttiin tunnistamaan useampia. Seuraavassa on tehty pohdintaa kuvassa 1 esitetyistä haasteista.

Yhtenä keskeisenä käytettävyyden kehittämisen haasteena, näytti aiemmissa julkaisuissa painottuvan ohjelmistojen kehittäminen teknisesti osaavammille käyttäjille. Aikaisempien julkaisujen perusteella avoimen lähdekoodin kehitystä toteutetaan omiin tarpeisiin (O'Reilly, 1999; Zhao & Elbaum, 2002) ja kehittäjiltä kehittäjille (Benson et al., 2004). Täten voidaan olettaa, että teknisesti osaamattomampien käyttäjien tarpeet eivät ole aina kehityksen keskiössä. Julkaisujen mukaan perinteiset käytettävyys ongelmat nousevat esiin usein juuri siinä vaiheessa, kun ohjelmistoja aletaan suunnittelemaan osaamattomammille käyttäjille. Näin ollen ohjelmistojen kehittäminen teknisesti osaavammille käyttäjille, asettaa haasteen loppukäyttäjän käytettävyyden kehittämiselle. (Nichols & Twidale, 2003.)

Toinen haaste, jolle aiempi tutkimus näytti antavan painoarvoa, oli käytettävyyden ammattilaisten puuttuminen projekteista. Käytettävyyden argumentoidaan olevan asia, jonka asianmukainen kehitys vaatii käytettävyyden ammattilaisten panostusta (Zhao & Deek, 2005). Aiempi tutkimus viittaasi kuitenkin siihen, että avoimen lähdekoodin kehitykselle on tyypillistä, että kehityksessä ei ole mukana käytettävyyden ammattilaisia (Benson et al., 2004; Nichols & Twdiale, 2003).

Resurssien puute itsessään nousi aiemmassa tutkimuksessa esiin merkittävänä haasteena. Sen voidaan olettaa olevan merkittävä haaste siinäkin mielessä, että se näyttää olevan yksi osasyy toiseen isoon haasteeseen, käytettävyyden ammattilaisten puutteeseen. Yhtensä keskeisenä syynä resurssien puutteeseen aiempi tutkimus painotti sitä, että perinteisten avoimen lähdekoodin projektien taustalla on harvoin kaupallisia toimijoita (Hertel et al., 2003; Nichols & Twidale, 2006).

Avoimen lähdekoodin yhteisöjen kulttuuri ja asenteet vaikutti jääneen tunnistetuista haasteista vähimmälle huomiolle aiemmissa tutkimuksissa. Ne nousivat kuitenkin esille yhtenä käytettävyyden kehittämisen haasteina, joten ne voisivat tarjota tutkittavaa vielä jatkossakin. Kulttuurin kuvaamisessa viitattiin paljon hakkerikulttuuriin (Madey, Freeh, Tynan, 2002), josta avoimen lähdekoodin kehitys juontaa juurensa. Aiemmassa tutkimuksessa painottui myös avoimen lähdekoodien yhteisöjen sulkeutuneisuus (Nichols & Twidale, 2003; Nichols et al., 2001). Samoin kun resurssien puute, myös kulttuuri ja asenteet vaikuttavat nousevan aiemmissa tutkimuksia esiin yhtenä mahdollisena osatekijänä käytettävyyden ammattilaisten puutteeseen.

6. Yhteenveto

Avoimen lähdekoodin käyttö ohjelmistokehityksen työkaluna on kasvattanut yhtä enenevissä määrin suosiotaan (Nichols, Thomson & Yates, 2001). Käyttäjille on tänä päivänä tarjolla lukuisia onnistuneita avoimen lähdekoodin ohjelmistoja (Nichols & Twdiale, 2003), jotka pystyvät toiminnallisuudellaan haastamaan samankaltaisia, kaupallisten ohjelmistoyritysten tarjoamia omisteisia ohjelmistoja (Andreasen, Nielsen, Schrøder & Stage, 2006). Avoimen lähdekoodin ohjelmistoja on kuitenkin kritisoitu niiden heikommasta käytettävyydestä (Andreasen et al., 2006).

Avoimen lähdekoodin ohjelmistojen käytettävyyden tutkiminen on noussut ajankohtaiseksi aiheeksi, johtuen ohjelmistojen leviämisestä (Zhao & Deek, 2005). Leviämisen myötä ohjelmistot ovat saaneet lisää teknisesti osaamattomampia käyttäjiä (Frishberg, Dirks, Benson, Nickell & Smith, 2002). Tämä on lisännyt tarvetta avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen, sillä perinteisten käytettävyysongelmien on havaittu nousevan esiin juuri siinä vaiheessa, kun ohjelmistoja aletaan suunnittelemaan teknisesti osaamattomammille käyttäjille (Nichols & Twidale, 2003).

Tämän tutkielman tavoitteena oli tunnistaa keskeisiä käytettävyyden kehittämisen haasteita, joita avoimen lähdekoodin ohjelmistojen kehitykseen liittyy. Lisäksi tarkoituksena oli löytää ohjelmistojen kehitysprosessista syitä, jotka vaikuttavat näiden haasteiden syntymiseen. Tutkielma toteutettiin perehtymällä aiheesta aiemmin tehtyihin julkaisuihin ja tutkimuksiin.

Tutkielman löydöksinä onnistuttiin tunnistamaan neljä keskeistä haastetta, joilla havaittiin olevan vaikutuksia avoimen lähdekoodin ohjelmistojen käytettävyyden kehittämiseen. Nämä neljä haastetta olivat: ohjelmistojen kehittäminen teknisesti osaavammille käyttäjille, käytettävyyden ammattilaisten puuttuminen, resurssien puute ja kulttuuri ja asenteet.

Ohjelmistojen kehittämiseen teknisesti osaavammille käyttäjille, tunnistettiin useampia mahdollisia syitä. Avoimen lähdekoodin kehitystä toteutetaan usein omiin tarpeisiin (O'Reilly, 1999; Zhao & Elbaum, 2002) ja kehittäjiltä kehittäjille (Benson et al., 2004), joten osaamattomampien käyttäjien tarpeet voivat jäädä pienemmälle huomiolle. Löydösten mukaan avoimen lähdekoodin ohjelmistot epäonnistuvat loppukäyttäjän kannalta käytettävyydessä siinä, että käytettävyysongelmia katsotaan niin sanotusti väärin silmin, jonka vuoksi ne jäävät huomaamatta (Nichols & Twdiale, 2003).

Käytettävyyden ammattilaisten puutteeseen löydettiin myös useampia mahdollisia tekijöitä. Yhtenä näistä tunnistettiin resurssien puute, jonka takia avoimen lähdekoodin yhteisöillä ei ole mahdollisuutta palkata käytettävyysammattilaisia kehitykseen (Nichols & Twidale). Avoimen lähdekoodin yhteisöt voivat olla myös haastava paikka työskennellä käytettävyyden ammattilaisina, johtuen projektien insinöörivetoista luonteesta, joka voi olla ristiriidassa käytettävyyden suunnittelumenetelmien kanssa (Benson et al., 2004). Muina mahdollisina tekijöinä käytettävyysammattilaisten puutteeseen, näyttäytyi yhteisöjen sulkeutuneisuus (Madey et al., 2002; Nichols et al., 2001; Nichols & Twdiale, 2003), käytettävyysammattilaisten vähäinen määrä suhteessa kehittäjiin, kiinnostuksen puute ja perinteet siinä mielessä, että avoimen lähdekoodin kehityksessä ei ole aiemmin ollut tarvetta käytettävyyden ammattilaisille (Nichols ja Twidale, 2003).

Resurssien puutteen selittäjänä, näyttäytyi pitkälti kaupallisten toimijoiden puuttuminen. Perinteisten avoimen lähdekoodin projektin taustalla on harvoin minkäänlaista instituutiota tai kaupallista toimijaa, joka maksaisi vapaaehtoisesti työskenteleville kehittäjille palkkaa heidän tekemästään työstä (Hertel et al., 2003; Nichols & Twidale, 2006).

Kulttuurin ja asenteiden tuomien haasteiden taustalla, näyttäytyi paljolti hakkerikulttuuri Madey, Freeh, Tynan, 2002), josta avoimen lähdekoodin yhteisöjen on mainittu ammentavat voimansa. Tällainen kulttuuri voi näyttäytyä ulkopuolisille henkilöille hyvinkin suljetuiksi, ja muualta kuin kehittäjien keskuudesta tuleva panostus ohjelmistojen kehittämiseen on usein pientä. (Nichols & Twidale, 2003; Nichols et al., 2001.) Yhteisöjen asenteista voi kertoa mahdolliset stereotypiat, joita avoimen lähdekoodin kehittäjillä voi olla käytettävyysammattilaisia kohtaan. Heidät saatetaan nähdä henkilöinä, joiden tehtävänä on vain viilailla käyttöliittymäelementtien välisiä etäisyyksiä ja suunnitella kaikki näyttämään samalta. (Frhisberg et al., 2002.) Kulttuuri ja asenteet tunnistettiin aiheena, joka voisi tarjota vielä paljon tutkittavaa tulevaisuudessa.

7. Lähteet

Andreasen, M. S., Nielsen, H. V., Schrøder, S. O., & Stage, J. (2006). Usability in open source software development: Opinions and practice. *Information technology and control, 35*(3).

Benson, C., Muller-Prove, M., & Mzourek, J. (2004). Professional usability in open source projects: GNOME, OpenOffice.org, NetBeans. *Proceedings of the CHI '04 Extended Abstracts on Human Factors in Computing Systems*, 1083-1084.

Borgholm, T., & Halskov, M. (1999). Cooperative usability practices. *Communications of the ACM, 42*(5), 91-97.

Botella, F., Alarcon, E., & Peñalver, A. (2014). How to classify to experts in usability evaluation. *Proceedings of the XV International Conference on Human Computer Interaction,* 1-4.

Frishberg, N., Dirks, A. M., Benson, C., Nickell, S., & Smith, S. (2002). Getting to know you: Open source development meets usability. *Proceedings of the CHI '02 Extended Abstracts on Human Factors in Computing Systems,* 932-933.

Hertel, G., Niedner, S., & Herrmann S. (2003). Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux kernel. *Research Policy, 32*(7), 1159-1177.

Hollingsed, T., & Novick, D. G. (2007). Usability inspection methods after 15 years of research and practice. *Proceedings of the 25th annual ACM international conference on Design of communication*, 249-255.

Iivari, N., Rajanen, M., & Hedberg, H. (2014). Encouraging for enculturation–An enquiry on the effort of usability specialists entering OSS projects. *Proceedings of the 25th Australasian Conference on Information Systems.*

International Organization for Standardization. (1998). 9241-11:1998 Ergonomic requirements for office work with visual display terminals (VDTs) — Part 11: Guidance to usability. Lainattu 20.1.2016, saatavilla: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-1:v1:en

Karels, M. J. (2003). Commercializing open source software. *Queue, 1*(5), 46-55. Lainattu 12.1.2016, saatavilla: http://queue.acm.org/detail.cfm?id=945125

Madey, G., Freeh, V., & Tynan, R. (2002). The open source software development phenomenon: An analysis based on social network theory. *Proceedings of the Eighth Americas Conference on Information Systems (AMCIS),* 1806-1813.

Nichols, D. M., Thomson, K., & Yeates S. A. (2001). Usability and open-source software development. *Proceedings of the Symposium On Coputer Human Interaction*, 49-54.

Nichols, D. M., & Twidale, M. B. (2003). The usability of open source software. *First Monday, Volume 8, number 1 – 6*, Lainattu 29.10.2015, saatavilla: http://firstmonday.org/ojs/index.php/fm/article/view/1018/939

Nichols, D. M., & Twidale, M. B. (2006). Usability processes in open source projects. *Software Process: Improvement and Practice*, *11*(2), 149-162.

Nielsen, J. (1993). *Usability engineering*. San Francisco, CA; Morgan Kaufmann.

Nielsen, J. (1994). Usability inspection methods. *Proceedings of the Conference companion on Human factors in computing systems,* 413-414.

Oracle. (2010). Oracle completes acquisition of Sun [Press Release]. Lainattu 20.1.2016, saatavilla: http://www.oracle.com/us/corporate/press/044428

O'Reilly, T. (1999). Lessons from open source software development. *Communications of the ACM, 42*(4), 32-37.

Pemberton, S. (2004). Scratching someone else's itch: (why open source can't do usability. *Interactions, 11*(1), 72.

Rajanen, M., Iivari, N., & Anttila, K. (2011). Introducing usability activities into open source software development projects - Searching for a suitable approach. *Journal of Information Technology Theory and Application (JITTA), 12*(4), 5-26.

Rajanen, M., Iivari, N., & Keskitalo, E. (2012). Introducing usability activities into open source software development projects - A participative approach. *Proceedings of the 7th Nordic Conference on Human-Computer Interaction (NordiCHI 2012),* 683-692.

Rajanen, M., & Iivari, N. (2013) Open source and human computer interaction philosophies in open source projects - Incompatible or co-existent? *Proceedings of the Academic MindTrek 2013.*

Raymond, E. (2001). *The cathedral & the bazaar: Musing on Linux and open source by an accidental revolutionary knowledge*. Sebastopol, CA: O'Reilly Media, Inc.

Robweir. (2012). OpenOffice.org is now Apache OpenOffice. [Web blog post]. Lainattu 20.1.2016, saatavilla: https://blogs.apache.org/OOo/?page=4

Rosenbaum, S. (1989). Usability evaluations versus usability testing: When and why? *Professional Communication, IEEE Transactions on*, *32*(4), 210-216

Trudelle, P. (2002). Shall we dance? Ten lesson learned from Netscape's flirtation with open source UI. Lainattu 17.12.2015, saatavilla: http://www.iol.ie/~calum/chi2002/peter_trudelle.txt

Von Krogh, G., & Von Hippel, E. (2003). Special issue on open source software development. *Research Policy, 32*(7), 1149-1157.

Zhao, L., & Deek, F. P. (2005). Improving open source software usability. *Proceedings of the Eleventh Americas Conference on Information Systems*, 923-928.

Zhao, L., & Elbaum, S. (2002). Quality assurance under the open source development model. *Journal of System and Software, 66*(1), 65-75.