Ritva Hartama-Heinonen, Marja Kivilehto, Liisa Laukkanen & Minna Ruokonen (toim./red./eds) 2017
MikaEL
Kääntämisen ja tulkkauksen tutkimuksen symposiumin verkkojulkaisu, vol. 10, s. 88–100
Nätpublikation för Symposiet om forskning i översättning och tolkning KäTu, vol. 10, s. 88–100
Electronic Journal of the KäTu Symposium on Translation and Interpreting Studies, Vol. 10, p. 88–100

Joukkoistetun kotoistamisen luonteesta kääntämisenä

Tommi Nieminen
Itä-Suomen yliopisto, Joensuu

Abstract

Web 2.0 has led to proliferation of crowdsourcing in many areas, not the least of which is translation and particularly localization. Another, simultaneous but unconnected phenomenon has been the proliferation of free/libre open source software (FLOSS). We use examples from a case of crowdsourced localization of FLOSS, the Finnish localization of KDE Plasma desktop environment. We discuss the role of the translator in FLOSS, the nature of crowdsourced localization as translation, and the ways in which crowdsourced localization may change the way we see some basic concepts of translation such as the source and target texts.

Keywords: crowdsourcing, localization, free/open source software, Web 2.0

Avainsanat: joukkoistaminen, kotoistaminen, vapaat ohjelmistot, Web 2.0

1 Johdanto

Pohdin tässä artikkelissa vapaiden tietokoneohjelmien joukkoistetun kotoistamisen luonnetta käyttäen aineistonani vapaan KDE- tai nyttemmin KDE Plasma -työpöytäympäristön suomalaista kotoistusta. KDE Plasma on vapaa (luvussa 2 esitetyn määritelmän mukaisesti) graafinen työpöytäympäristö, joka tarjoaa paitsi monipuolisen työpöytäympäristön myös laajan hyöty- ja viihdeohjelmakokonaisuuden. Sen tyypillisin käyttöympäristö on Linux, mutta se on saatavilla myös muille alustoille. Esimerkkini ovat kotoistamisen yhteydessä kohtaamistani todellisista ongelmista tai haasteista, joskaan kaikkia ei ole tässä artikkelissa mahdollista käydä järjestelmällisesti läpi.

Tietokoneohjelmat ovat nykyajan keskeisimpiä työympäristöjä, ja internet on muuttanut niiden toimintaa merkittävästi viime vuosikymmeninä. Ensi vaiheessa verkkoon kytkeytyivät vain erityistarkoituksiin luodut ohjelmat, kuten selain, sähköposti ja tiedonsiirto, jolloin käyttäjä helposti tunnisti rajapinnan koneensa ja ulkomaailman välillä. Nyttemmin lähes jokainen ohjelma toimii verkossa ja verkosta on tullut niin keskeinen osa tietokoneen toimintaa, että verkoton tietokone alkaa olla käsitteellinen anomalia. Suomalaisen nimensä mukaisesti tietokoneesta on kehkeytynyt **tieto**kone, ja tieto löytyy pilvestä. Verkko on muuttanut digitaalisen sisällöntuotannon kuvaa (Lewis ym. 2014: 99). Samalla ihmisten toiminta tietokoneen ääressä on globalisoitunut, mikä on synnyttänyt kasvavan tarpeen kääntämiselle, jotta eri puolilla maailmaa sijaitseva tieto ja

palvelut tuotaisiin käyttäjien ulottuville (O'Hagan 2012: 124; Fernández Costales 2013: 86–87) – ainakin kaikkien niiden, jotka eivät hyväksy ajatusta yhdestä maailmantyökielestä (vrt. Folaron 2012: 6).

Web 2.0:n myötä on toteutettu jo sen alkuperäiseen konseptiin kuulunut vuorovaikutteisuuden ajatus, joka on paitsi uudistanut tapoja luoda ja välittää ohjelmia myös johtanut yhä pienempien ohjelmistoyritysten kansainvälistymiseen (Bell & Loane 2010: 213–214). Alkuperäinen 1990-luvulla syntynyt WWW (Web 1.0) oli yksisuuntainen: verkon aineisto oli rakenteista tekstiä tai koneelle ladattava binaarimöykky (engl. *binary blob*). Aineiston toimitti joku verkkoon, mistä käyttäjät sen sellaisenaan noutivat. Web 2.0:ssa käyttäjätkin nähdään voimavarana, joka voidaan osallistaa aineiston tuottamiseen (Collis & Moonen 2008: 94; Fernández Costales 2013: 88; Tung & Tseng 2013: 2143). Tämä vaikuttaa myös kääntämiseen, joka nykyisellään muutenkin on digitaalista, ulkoistettua ja projektivetoista (Dunne 2012: 144). Monet kääntämisen osa-alueet, kuten kotoistaminen (esim. Mesipuu 2012) ja AV-kääntäminen (esim. Munday 2012), siirtyvätkin helposti ja luontevasti Web 2.0 -aikaan, etenkin kun näiden kahden keskinäinen suhde on muutoksen myötä jo sumennut (O'Hagan 2012: 127).

Kotoistamisen joukkoistamisessa on kyse voitoista muttei vain kustannussäästöjen mielessä, kuten epäluuloisimmat ovat esittäneet (Nogueira & Semolini 2010). Joukkoistaminen mahdollistaa myös kokonaan uusia asioita: missä aiemmin vain suurimmat kaupalliseen menestykseen tähtäävät ohjelmat oli taloudellisesti mahdollista kotoistaa, nykyään kotoistus ulottuu pieniin harrastajaprojekteihinkin, joilla kaikilla on näin mahdollisuus kansainväliseen saatavillaoloon (Bell & Loane 2010: 214–215). Kääntäminen ja kotoistaminen ovat tärkeässä osassa globalisaatiossa ja kaupallisessa(kin) menestyksessä (Pettini 2015: 277). Pelkästä viestintävälineestä internet on kasvanut yhteistoimintatasoksi, joka ohjaa ohjelmakehitystyötä uusiin suuntiin (Bell & Loane 2010: 218). Kääntämisestä tulee yleisöpalvelu, joka mahdollistaa yhteistoiminnan eri muodot (Folaron 2012: 26).

Muokattavuus ja muunneltavuus ovat Web 2.0:n keskiössä, mikä toisaalta haastaa esimerkiksi perinteisen asiantuntijoiden takaaman laadun käsitteen (Collis & Moonen 2008: 95, 103). Aineisto ei enää saavu käyttäjälle valmiina, asiantuntijoiden loppuun saakka hiomana, vaan aihiona, josta muokata kokonaisuus omiin käyttötarpeisiin.

Toistaiseksi harvalukuisissa tutkimuksissa käyttäjien on havaittu suhtautuvan työn joukkoistamiseen viihteenä tai pelinä, joka tarjoaa heille tavan toteuttaa itseään ja kokea tyydytystä tai saada tunnustusta osaamisestaan (Estellés-Arolas & González-Ladrón-de Guevara 2012: 194–195; O'Hagan 2012: 125; Djelassi & Decoopman 2013: 687). Joukkoistaminen on osa laajempaa yhteisöllisen toiminnan kontekstia, jossa kontribuutiot kulkeutuvat käyttäjiltä toisille käyttäjille vuorovaikutuksessa, joka parhaimmillaan hyödyttää kaikkia osapuolia. Aktiivisten toimijoiden motivaatio vaihtelee altruistisesta omien taitojen kehittämiseen, ja mukana on harrastelijoiden lisäksi ammattilaisia (Fernández Costales 2013: 92–93). Vaarana on työn tuottaman arvon valuminen yrityksille, jotka hyödyntävät yhteisöjä (lähes) kustannuksettomana osana toimintaansa (mas. 95). Tähän pyritään vastaamaan tietokoneohjelmien vapaudella (ks. alalukua 3).

Pohdin artikkelissani seuraavia kysymyksiä:

1. Mikä on kotoistajan rooli vapaiden ohjelmistojen liikkeessä?
2. Millainen on vapaiden ohjelmistojen joukkoistetun kotoistuksen erityisluonne käännöstyönä?
3. Miten joukkoistettu kotoistaminen muuttaa suhtautumista kääntämisen lähtö- ja kohdetekstiin?

Kysymykset ovat luonteeltaan laajoja ja vaatisivat omat tutkimuksensa, mutta kokoan tässä yhteen havaintojani ja käsityksiäni useamman vuoden KäTu-esitelmieni pohjalta jatkotarkastelua varten.

2 Käsitteitä

Kotoistaminen eli **lokalisointi** on toimintaa, jolla tuote, kuten tietokoneohjelma, pyritään sovittamaan paitsi kieleltään, myös muulta toiminnaltaan kunkin kohdeyleisön tarpeisiin sopivaksi. Kielellisten elementtien lisäksi muutoksia vaativat tyypillisesti merkistöt, aakkostus, ajan ilmaukset jne. Useimmat näistä mahtuvat kääntämisen perinteisiin rajoihin, joten kääntämisen ja kotoistamisen ero on pikemmin näkökulmassa; tässä artikkelissa hahmottelen, miltä tilanne kotoistajan silmin näyttää.

Vapaalla ohjelmistolla tarkoitetaan ohjelmaa, jonka lähdekoodi on vapaasti käyttäjän saatavilla ja muokattavissa. Vapaus ei suoraan ota kantaa kustannuksiin tai rahan siirtymiseen. Yksi vapaiden ohjelmistojen liikkeen keskeinen toimija, Free Software Foundation (FSF, 1986), luettelee neljä keskeistä piirrettä:

1. vapaus käyttää ohjelmaa mihin tarkoitukseen tahansa
2. vapaus tutkia ohjelman toimintaa ja soveltaa sitä omiin tarpeisiin
3. vapaus levittää edelleen ohjelman kopioita
4. vapaus parannella ohjelmaa ja julkaista tehdyt muutokset.

On syytä korostaa, ettei FSF viittaa sanalla *free* ’ilmaisuuteen’ vaan ’vapauteen’. Joskus ero täsmennetään sanaparilla *free*/*libre*. Ohjelma voi olla maksullinen ja silti vapaa; merkitsevää on vain se, saako käyttäjä ohjelman lisäksi myös sen koostamiseen käytetyn lähdekoodin, jotta hänen vapautensa muuttaa ohjelmaa säilyisi.

Piirteiden takaista ideologiaa ohjaavat määräperiaatteet. Ensinnäkin tietokoneen halutaan voimavarana kuuluvan kaikille. Näin pyritään takaamaan ihmisyhteisön kulttuurisen pääoman kuuluminen kaikille sekä jokaisen mahdollisuus oppia jo tehdystä. Toisekseen vapaus takaa, ettei kenenkään tarvitse tyytyä valmiisiin vaihtoehtoihin. Pääsy lähdekoodiin takaa käyttäjälle mahdollisuuden räätälöidä ohjelmaa tarpeisiinsa. Tätä ohjaa ajatus, ettei kukaan tunne ennalta kaikkien tarpeita. Alhaalta ylös -ajattelu mahdollistaa kaikkien osallistumisen kehitystyöhön eri tavoin, kunkin kykyjensä mukaan. (Ks. esim. Stallman 2009.)

Joukkoistaminen ja **talkoistaminen** ovat englannin sanan *crowdsourcing* nyttemmin vakiintuneita suomalaisia käännösvastineita. Englannin uudissana muodostettiin sanoista *crowd* 'väkijoukko' ja *outsource* 'ulkoistaa' (tai yhden näkemyksen mukaan yleisemmästä verbistä *source* 'hankkia jokin jostakin', Estellés-Arolas & González-Ladrón-de Guevara 2012: 189). Merkitys rakentuu osistaan: kyse on toiminnan osan ulkoistamisesta määrittämättömälle toimijajoukolle. Joukkoistamisesta on uhannut tulla muotisana, joka hautaa alleen muita yhteistoiminnan muotoja (mas. 189), mutta tässä yhteydessä tarkoitan sitä sanan varsinaisessa mielessä: jokin toimija, kuten tietokoneohjelman kehittäjä, ulkoistaa jonkin osan toteuttavaa kokonaisuutta kenen tahansa vapaasti mutta maksutta toteutettavaksi. Suomalaisista käännösvastineista pidän vapaiden ohjelmistojen kotoistamiseen sopivimpana joukkoistamista syistä, jotka käyvät tarkemmin ilmi luvusta 3; lyhyesti on kyse siitä, että vapaissa ohjelmistoissa kotoistus toteutetaan oleellisesti samoin kuin koko muu tuotanto eikä kyseessä ole vain jonkin osa-alueen toteuttaminen talkoilla.

Web 2.0:lla tarkoitetaan lyhyesti kerraten paitsi välineitä, joilla verkon yli on mahdollista toimia yhä vuorovaikutuksellisemmin, myös sitä uudentyyppistä tiedon ja toiminnan mallia, jonka tällaisten välineiden olemassaolo saa aikaan. Web 2.0:n ja sen mahdollistaman joukkoistamisen merkityksen katsotaan ulottuvan ohjelmistokehityksestä aina tietojärjestelmien käsitteeseen (Majchrzak & Malhotra 2013) ja tieteenteon muutokseen (Lievrouw 2010) saakka. Web 2.0:aan liittyy kansainvälistyminen ja globalisoituminen, jotka, kuten todettua, synnyttävät myös lisääntyvää kääntämisen tarvetta.

3 Vapaiden ohjelmistojen kotoistus päättymättömänä projektina

Vapaiden ohjelmistojen ideologiassa passiivisen kuluttajan tilalle havitellaan aktiivista käyttäjä-kehittäjää. Lähdekoodin avoimuus takaa, että jokaisella on mahdollisuus suunnata jo olemassa olevia projekteja haluamaansa suuntaan.

Kotoistaja kuuluu tähän keskeisesti. Ensinnäkin kotoistus takaa pääsyn ohjelmaan yhä suuremmalle käyttäjäryhmälle. Toisekseen kotoistettu ohjelma ottaa toiminnoissaan huomioon useammanlaisten ihmisten tarpeet. Kotoistajakaan ei kuitenkaan saa yksinvaltaa työhönsä: ryhmä antaa ja ryhmä ottaa – jotkin ratkaisut muut hyväksyvät, jotkin torjuvat, ja tähän on tyydyttävä. Jos ryhmä esimerkiksi päättää siirtyä *viive : viiveen* -tyyppiseen, kielenhuollon nyttemmin (hieman käsittämättömästi) hyväksymään taivutuskaavaan, yksittäisen kääntäjän toive kielihistoriallisesti oikeammasta ja kielikorvaa hivelevämmästä *viipeen*-taivutuksesta ei mene läpi (kirjoittajan yksityinen sähköpostikirjeenvaihto). Tulokset siirretään yhteiseen pääomaan. Kotoistajasta tulee myös aktiivinen osatoimija, käyttäjä-kehittäjä, jonka on ratkaistava kantansa siihen, mitä kehitystyön "ylävirrassa" tapahtuu. Huonoimmillaan tämä jää reagoinniksi ulkoisiin tapahtumiin, mutta parhaimmillaan kotoistajat toimivat aktiivisena osana kehitystyötä ja voivat vaikuttaa siihen, jos se ajautuu ristiriitaan kotoistettavuuden kanssa.

Kääntäjät ovat yleensä suhtautuneet joukkoistettuun kääntämiseen torjuvasti (esim. Nogueira & Semolini 2010). Se on nähty ammattitaidottomien harrastelijoiden omaehtoiseksi mutta ajattelemattomaksi toiminnaksi, jota häikäilemättömät yritykset käyttä-

vät hyväkseen. Joukkoistettuun kääntämiseen osallistuvista tiedetään toistaiseksi vähän, mutta Olohan (2014: 20–21, 27) on osoittanut, että osallistumisen motiivit ovat sekä yhteisöllisiä että yksilöllisiä, joista yhteisölliset tapaavat painottua itse asiassa enemmän. Joukkoistettu kääntäminen nähdään yleisöpalveluksi tilanteessa, jossa vaihtoehtona on vain kääntämättömyys. Vapaat ohjelmistot ovat selkeästi tällainen tilanne, koska tyypillisesti niitä levitetään paitsi vapaasti myös ilmaiseksi: raha ei liiku käyttäjiltä kehittäjille sen enempää kuin kehittäjiltä kotoistajillekaan.

Vapaa ohjelmisto on luonteeltaan vapaasti kotoistettavissa. Kääntäminen tapahtuu yhteisöissä, joissa ohjelman kehittäjillä ei ole erityistä valtaa. Yhteisöjen sisällä valta puolestaan voi olla täysin tasan jakautunutta tai ns. meritokraattista, ansioihin perustuvaa (Weber 2004: 144–145): yhteisö arvioi kunkin osallistujan merkityksen hänen läsnäolonsa keston ja työnsä laadun perusteella, mikä tutkimustenkin perusteella patistaa vapaaehtoisen toimijan myös tekemään työtään laadukkaammin (O'Mahony & Ferraro 2007: 1082). Vapaiden ohjelmistojen kotoistaminen kuuluu Fernández Costalesin (2013: 95–97) luokituksessa pikemmin yhteistoiminnalliseen (”fani-”) kuin joukkoistettuun kääntämiseen. Parhaimmillaan kotoistajien ja kehittäjien suhteesta tulee molempisuuntainen ja kumpaakin osapuolta hyödyttävä, kun tieto kotoistamisen toiveista tai ongelmista voi välittyä eteenpäin kehittäjille.

Yksinkertainen esimerkki vuorovaikutustarpeesta on käyttöliittymän säädin, jossa odotetaan käännöstä pelkästään englannin prepositiosta ”To” koostuvalle selitteelle, jota seuraa toinen säädin (esim. numero- tai päivämääräsäädin). Suomenkielisessä käyttöliittymässä tällaisen selitteen kääntäminen on ongelmallista, vaikka olisikin tiedossa, millainen graafinen elementti tekstiselitettä seuraa. Kääntäjä voi kaivata esimerkiksi vaihtoehtoista selitettä säätimen jälkeen postpositiota varten tai koko graafisen kokonaisuuden uudelleensuunnittelua. Juuri näitä toiveita kehittäjien suuntaan on toistuvasti tarpeen välittää. Yleisemminkin on havaittu ammattikehittäjien – joiksi vapaan ohjelmistojen kehittäjät voi vähintään tässä kontekstissa lukea – tuottavan ideoita, jotka ovat vähemmän innovatiivisia mutta yleensä yksinkertaisempia ja nopeampia toteuttaa kuin ne, jotka tulevat joukkoistetuilta maallikoilta (Poetz & Schreier 2012: 251), joiksi tässä kontekstissa lukeutuvat myös kotoistajat. Ongelma heijastuu ohjelmien kotoistettavuuteen.

Monet vapaat ohjelmat, kuten KDE Plasma, eivät tee jyrkkää rajaa piilossa tapahtuvan kehitystyön ja lopullisen julkistamisen välillä, joten kotoistuksessa on lähes pysyvästi se tilanne, että käännös laahaa kulloisenkin kehitysvaiheen perässä. Osin tämä johtuu myös edellä mainitusta vastavuoroisesta suhteesta kotoistusryhmien ja kehittäjien välillä, koska ryhmiä on useita ja niiden kehittäjille välittämät toiveet keskenään tahdistamattomia. Lähdekoodiin voidaan toisin sanoen tehdä yhden kotoistajan toiveesta juuri sellaisia muutoksia, jotka toisen kielen kannalta johtavat joko turhaan uudelleenkääntämistarpeeseen tai ovat jopa haitallisia. Perinteisen yritysmaailman suljetussa kehityskulttuurissa tuote saapuu käännettäväksi (enemmän tai vähemmän) käännösvalmiina, jolloin käännöksen on mahdollista ehtiä kokonaisuudessaan mukaan julkistettuun tuotteeseen. Vapaissa ohjelmissa jokainen vaihe on (jälleen: enemmän tai vähemmän) julkinen, ja kotoistus tapahtuu yleensä nopeasti muuttuvassa kehityshaarassa. Siten on tavallista, että ahkerastakin kotoistuksesta huolimatta loppukäyttäjän eteen saapuu puolikielinen työpöytäympäristö: osa käännöksistä ei ehdi mukaan, osa koskee vasta

tulossa olevaa julkaisuversiota, osa on jo ohjelmakehitystyön myötä automaattisesti merkitty vanhentuneiksi eivätkä ne näy käyttäjälle, vaikka ne sinänsä olisivat täysin hyväksyttäviä. Käännösprojekti jää pysyvään keskeneräisyyden tilaan.

Suuri osa ohjelmissa tapahtuvista muutoksista liittyy niiden oman toiminnallisuuden kehittymiseen kohti jotakin jo ennalta olemassa olevaa tavoitetta. Esimerkiksi kuvankäsittelyohjelmassa kotoistaja tietää jo suunnilleen ennalta, millainen ohjelma tulee olemaan saavutettuaan toiminnallisuudessa tavoitteensa, koska kuvankäsittelyohjelmia on ollut olemassa ennenkin ja ne antavat mallin uudellekin ohjelmalle. Ohjelman uudet piirteet eivät tule täytenä yllätyksenä vaan niihin on mahdollista periaatteellisesti varautua etukäteen. Systeemis-funktionalistisen kielitieteen merkityspotentiaalin käsitettä soveltaakseni (esim. Halliday 1978: 39–40) on ikään kuin **käännöspotentiaali**, josta ohjelman kehittyessä toteutuu yhä suurempi osuus. Käännöspotentiaalin muodostaisivat näin kaikki mahdolliset käännöstekstit ja -jonot, jotka annetussa merkitysten avaruudessa on mahdollista tuottaa. Yhä jää mahdolliseksi, että uusi, vasta kehitteillä oleva ohjelma kehittyy kaikkien aiempien malliensa ohi, jolloin potentiaali kasvaa täysin uuteen ja ennakoimattomaan suuntaan. Tällöin kotoistajakin on täysin tuntemattomilla vesillä ja joutuu mahdollisesti huomaamaan myös kohdekielen sanastosta puuttuvan tarvittavia sanoja.

Silloinkin kun kyse ei ole vain toisen ohjelman toiminnallisuuden toisintamisesta, ohjelmien kehityksessä voi havaita kääntäjän työtä helpottavaa asteittaisuutta. Tietokoneympäristöt tapaavat olla sikäli evolutiivisía, etteivät uudet vaiheet yleensä täysin hylkää aiempien saavutuksia. Nykyisissä, 10–20 vuoden takaisista jo merkittävästi eroavissa tietokoneen käyttöliittymissä valtaosa elementeistä palautuu aiempiin malleihin, joita ovat antaneet laajalti sellaiset järjestelmät kuin vuonna 1984 julkaistu Applen Mac-käyttöjärjestelmä (tuolloiselta nimeltään System), 1987 julkaistu NeXT, 1995 julkaistu Microsoft Windows 95 tai kaikkien yhteinen esiäiti, Xeroxin jo 1970-luvulla kehittämä kokeellinen graafinen käyttöliittymä.

Esimerkiksi toistuvista elementeistä käy KDE Plasman *paneeli* (engl. *panel*), joka nykymuodossaan ulottuu Windows 95:een, vaikka sen esikuvana oli jo NeXT. Paneeli on näytön alue, joka sallii käyttäjän esimerkiksi käynnistää uusia ohjelmia tai siirtää kohdistus ohjelmasta toiseen. Paneelista löytyy *tehtävänhallinta* (engl. *task manager*), joka samoin kuin *sovelluskäynnistin* (*application launcher*) eli kansanomaisemmin *Käynnistä-valikko* periytyvät Windows 95:stä; sovelluskäynnistin tunnettiin KDE:ssa aiemmin *K-valikkona*. Mahdollisuus lisätä paneeliin *pikakuvakkeita* (*shortcut*) käynnistettäviin muttei vielä käynnissä oleviin ohjelmiin periytyy NeXTiin asti. Windows 95:n peruja paneelissa on *ilmoitusalue* (*system tray* tai *notification area*), joka näyttää käyttäjälle pysyväluonteista tilatietoa tai sallii pääsyn joihinkin järjestelmän palveluihin (kuten verkon hallintaan, äänenvoimakkuuden säätöön, näppäimistöasettelun vaihtoon).

Kaikki edellä mainitut käyttöliittymäelementit ovat omilla tuotenimillään osa mainittuja järjestelmiä mutta myös osa yhteistä toiminnan kulttuuria, jossa niille tarvitaan kotoistettu kuvaileva yleisnimi. Kääntäjän ongelmana on tällöin, missä määrin ohjelmistossa tyytyä erisnimenomaisiin nimiin (esim. *K-valikko*), missä määrin taas yrittää suuntautua yleisnimiin (*sovelluskäynnistin*), jotka ensi vaiheessa ovat väkisinkin uudissanoja mutta jatkossa voivat helpottaa kääntämistä myös parhaillaan käännettävänä olevan

ohjelmaprojektin ulkopuolella. Käytännössä kyse on uudissanaston luomisesta: kohdekieleen ei voi suhtautua vain olemassa olevana valmiina voimavarana vaan sen kehittämistä täytyy pitää kotoistuksen yhtenä tavoitteena.

Muutoksen hankalin muoto on se, jossa kotoistaja kohtaa jotakin käsitteellisestikin uutta. KDE-työpöytäympäristön versiossa 4 esiteltiin eräs tällainen, *aktiviteetti* (engl. *activity*). Aktiviteetti tarkoittaa tapaa, jolla koko käyttäjän näkemä työpöytäympäristö toimii. Sen oletusarvo on perinteinen työpöytäkansionäkymä, jossa työpöytä on joko enimmäkseen passiivinen tausta avattaville ikkunoille tai siinä näkyy tiedostojärjestelmän erityisen työpöytäkansion sisältö kuvakkeina; jälkimmäinen on tuttu niin Windowsista kuin Mac OS:stakin. Toisenlaisiakin aktiviteetteja on kehitetty, esimerkiksi ehkä parhaiten kosketusnäytöille soveltuva *Etsi ja käynnistä* (*Search & Launch*), jossa koko työpöytä muodostaa sovelluskäynnistimen. ”Aktiviteetti” on oikeastaan huono nimitys tälle toiminnallisuudelle. Sanavastineeseen päädyttiin KDE-kotoistajaryhmän sähköpostikeskusteluissa heinäkuussa 2010 lähinnä, koska sen ilmestyttyä käännettäviin käännösjonoihin kotoistajillakaan ei ollut vielä tarkkaa kuvaa siitä, millainen ominaisuus täsmällisesti ottaen on kyseessä ja kuinka laajalti se tulisi käyttöön (kirjoittajan yksityinen sähköpostikirjeenvaihto). Käytännössä aktiviteetti nousi kotoistajien neuvonpidossa esiin aluksi väliaikaisena vaihtoehtona, jolla jono saatiin käännetyksi. Kukaan ei ollut valintaan erityisen tyytyväinen, ja tarkoituksena olikin korvata sana paremmalla myöhemmin. Kun aikaa kuluu ja useita versioita kulkee ohi käännöksen muuttumatta, väliaikainen käännös pääsee kuitenkin kuin huomaamatta vakiintumaan ja sitä on yhä vaikeampi enää muuttaa, koska käyttäjät ovat tottuneet siihen.

Loppukäyttäjän kannalta huonokin sanavaihtoehto on kuitenkin yleensä vakiintuneena parempi. Se takaa, että esimerkiksi verkkohauissa on mahdollista saada mielekkäitä osumia. Haettavuus (”guuglattavuus”) on yksi kotoistamista luonnollisesti ohjaava normi (Nieminen 2011: 9–10), ja se tahtoo korostua vapaissa ohjelmissa, joissa kunnollisia ohjeita ei useinkaan ole saatavilla. Vallankaan ohjeita ei löydy kotoistettuna, koska kääntäjälle suuritöiset ohjetiedostot jäävät usein kotoistuksessa viimeiseksi, mikä selittyy sillä, että käytännössä dokumentoinnin kääntäminen edellyttää, että ohjelma kaikkine kielellisine käyttöliittymäelementteineen on jo käännetty. Helposti käy niin, että vain jotta ohjeiden kääntämisen voi edes aloittaa, käyttöliittymät on käännettävä ensin vaikka alustavamminkin, mikä taas myöhemmin helposti lukitsee tilanteen väliaikaiseksi aiottuun ratkaisuun.

Haettavuus suuntaa käännösjonojen käännöksiä helposti takaisinkäännettävyyden mahdollistavaan eli kohdekielen kannalta epäidiomaattiseen suuntaan. *Aktiviteetti* on tästäkin hyvä esimerkki, koska se on helppo kääntää taaksepäin englanniksi *activity*ksi, mikä lisää mielekkäitä hakuosumia, koska valtaosa verkossa olevasta tiedosta on englanniksi. Haettavuus- ja takaisinkäännettävyysnormit – jotka liittyvät yhteen – aiheuttavat osaltaan, että väliaikainen, raa’an oloinen käännös voi joissakin tapauksissa olla käytettävyysnäkökulmasta jopa paras, vaikkei se vastaa kääntämisen perinteisiä normeja ja arvoja eikä välttämättä tyydytä kääntäjien (tai käyttäjienkään) kunnianhimoa eikä esteettistä silmää. Ilmiö on seuraamuksiltaan kielipoliittinen, ja siihen on varmasti tulevaisuudessa kiinnitettävä yleisemminkin huomiota.

Ohjelmistojen kehityksessä kaksi päällekkäistä samanaikaista ilmiötä kumpikin vaikeuttavat kääntämistä. Toisaalta samoja tai läheisesti samankaltaisia ominaisuuksia kehitetään eri tahoilla eri nimillä, mikä kertoo kehittäjien halusta ”brändätä” toteutustapansa. Toisaalta kehitetään erilaisia ominaisuuksia niin, että käyttötavat erilaistuvat, jolloin kääntäjän kehittämä yleisnimikään ei enää voi auttaa käyttäjää ohjelman käytössä. Joka tapauksessa kotoistajan on päätettävä, annetaanko brändäyksen viedä kehitystä omaan suuntaansa, mikä voi olla kehittäjien toive mutta johtaa käyttäjien käyttökokemusten sirpaloitumiseen, vai yritetäänkö kehittää yhteistä sanastoa, mikä voi johtaa ristiriitaan kehittäjien toiveiden ja suunnitelmien kanssa mutta ainakin hetkellisesti palvella kohdekielistä käyttäjäkuntaa paremmin. Koska kehittäjien toive saattaa kertoa myös halusta murtaa vanha malli asteittain, ei käytännössä ole mahdollista aivan täsmällisesti päättää, mikä käännösstrategia olisi ”käyttäjälähtöisin”. Voi hyvin olla, ettei mikään ole.

Kotoistajan strategianvalinnan vaikeutta korostaa se, että hänen edessään ei koskaan näy kokonainen ”lähdeteksti” eli toiminnallinen malli siitä, miten kehittäjä ohjelmaansa on ajatellut käytettävän – nyt ja tulevaisuudessa, kun kaikki aiotut muutokset on toteutettu. Käännettävänä on kyllä käännösjonoja, joista lähde- ja kohdeteksti muodostuvat, mutta itse teksti ei ole suoraan tavoitettavissa. Tämä johtaa seuraavassa luvussa kuvattuun ongelmaan.

4 Mikä on kotoistuksen lähdeteksti?

Kuten Juliane House (2009: 4; lihavoinnit alkuperäiset, suomennos TN) muotoilee, ”Kääntäminen on prosessi, jossa **lähdetekstinä** tunnettu alkuperäinen teksti korvataan **kohdetekstiksi** kutsutulla tekstillä”. Ydinkäsitteenä on siis **teksti**. Kääntäminen kohdistuu siis merkityskokonaisuuteen, ja kohdetekstin on samoin oltava merkitykseltään koherentti kokonaisuus. Tämä on kääntämisen ”platonistinen olemus”, yritys tavoittaa mieli muodon takaa (Berman 2007 [1986]: 79–81).

Kotoistuksessa ”teksti” muodostaa metodologisen haasteen. Ei ole itsestään selvää, mitä sillä tässä kontekstissa tarkoitetaan. Tekstin voi kuvata merkityksen sulkeumaksi (Nieminen 2010: 79–80): se rajautuu kontekstiin, jossa toiminta muodostaa mielekkään kokonaisuuden; näin esimerkiksi kauppa-asioinnin keskustelua asiakkaan ja myyjän välillä voi pitää tekstinä, kun taas ostoskeskuksen jonkinpäiväisten diskurssien kokonaisuus ei ole teksti – paitsi jos siitä sellainen rakennetaan esimerkiksi fiktiossa. Tämä ajatus on nyt mahdollista siirtää kotoistuksen kontekstiin.

Jo yksittäisen tietokoneohjelman samastaminen tekstiin on haastavaa (Lehtovaara 2010: 26–39; Nieminen 2011: 11–12), ja käsillä olevassa tapauksessa kyseessä on useiden ohjelmien muodostama toiminnallinen kokonaisuus. Kotoistuskontekstin tekstikäsitteen eriävyydestä on huomautettu aiemminkin (Pym 2006); Pym tosin jättää epäselväksi, **mistä** kotoistuksen teksti eroaa. Selvää on ainakin, että kotoistuksen välittömänä kohteena on irrallisista merkkijonoista koostuvan tiedosto, jonka käännettävät merkkijonot sen paremmin kuin niiden muodostama fyysinen kokonaisuuskaan eivät ole tekstejä.

Esimerkkinä olevassa projektissa käytetty kotoistustekniikka on GNU gettext, jonka tiedostomuoto on yksinkertainen mutta osin ongelmallinen (ks. esim. Nieminen 2011: 3–5; Lewis ym. 2014: 105). Tiedosto ei alkuaan antanut kotoistajalle mitään tietoa käännösjonojen – tyypillisesti lausetta lyhyempien kielellisten katkelmien – konteksteista. Viime aikoina kontekstien kirjaaminen tiedostoon on lisääntynyt, mutta yhä ollaan kaukana kehittyneempien, esim. XML:ään pohjautuvien tekniikoiden kyvystä tuottaa oheistietoa kääntämistä varten, ja kirjaaminen on sittenkin kehittäjän muistamisen ja yhteistyöhalun varassa.

Otetaan esimerkki. Alla on katkelma Kate-muokkaimen kotoistustiedostoa (josta on poistettu esimerkin kannalta merkityksettömiä kommentteja). Näyte on itse asiassa varsin edustava esimerkki käännösjonojen tyydyttävästä kontekstoinnista. `msgid`-rivit ovat käännettäviä jonoja, `msgstr`-rivit tuottaa tai niitä muuttaa kotoistaja käännöstyön myötä. #-alkuiset rivit ovat kommentteja, joista `#:`-alkuiset tarkoitettu tiedoston automaattiseen jäsennykseen. `msgid_plural` ja `msgstr[n]` (missä $n \geq 0$) sallivat kääntää erikseen yksikkö- ja monikkomuotoiset lähdejonot, mahdollisesti muitakin, jos kielessä on useampia lukuja tai muuten vaihteleva lukutaivutus.

```
#: kateconfigdialog.cpp:123
msgid "&Delete unused meta-information after:"
msgstr "&Poista käyttämättömät metatiedot:"

#: kateconfigdialog.cpp:127
msgctxt "The special case of 'Delete unused meta-information after'"
msgid "(never)"
msgstr "(ei koskaan)"

#: kateconfigdialog.cpp:128
msgctxt "The suffix of 'Delete unused meta-information after'"
msgid " day"
msgid_plural " days"
msgstr[0] " päivän kuluttua"
msgstr[1] " päivän kuluttua"
```

`msgctxt`-rivit ovat kehittäjän kääntäjälle osoittamia kontekstointeja, joita ei käännetä mutta jotka auttavat käännöstyössä. Etenkin viimeinen jono osoittaa tämän tarpeellisuuden. Kontekstitta kääntäjä voisi aavistaa, että jonoa edeltää numeerinen kenttä, koska käännösjono annetaan erikseen yksikköä ja monikkoa varten. Sen sijaan hänen olisi mahdotonta tietää, millainen tekstiselite luvun ja sanan *päivä* ~ *päivää* -paria edeltää, koska käännösjonojen järjestys on periaatteessa mielivaltainen: tiedoston voi järjestää vaikka uusiksi muuttamatta sen käyttökelpoisuutta, koska `msgid`-jonot toimivat nimensä mukaisesti tunnisteina paikoista, joihin käännös tulee. Usein tosin käyttöliittymässä lähekkäin olevat jonot ovat lähekkäin myös tiedostossa, koska ohjelmoija on luonut jonkin käyttöliittymänäkymän yhtenä kokonaisuutena; tähän ei kuitenkaan voi luottaa. Kontekstiksi annettu ohje auttaa muotoilemaan tekstin niin, että se sisältää myös suomalaisen postposition kuluttua. Tämän jälkeen ongelmaksi jää enää, sopiiko ehdotettu jono tilaan, joka on lähtökielen perusteella voitu varata liian pieneksi.

Tämänkin esimerkin perusteella lienee jo selvää, ettei käännöstiedosto muodosta minkäänlaista mielekästä, ymmärrettävää tekstiä – se ei tuota jo mainitsemaani merkityk-

sen sulkeumaa. Kotoistajan on silti yhtä välttämätöntä tuottaa teksteihinsä koherenssi kuin kenen tahansa muunkin kääntäjän. Loppukäyttäjä näkee käännetyt jonot joinakin yhdistelminä joissakin konteksteissa, mutta kotoistajalla ei ole näihin konteksteihin suoraa pääsyä. Esimerkiksi yllä olevassa tiedostokatkelmassa jonot ”Poista käyttämättömät metatiedot” ja ”(ei koskaan)” tulevat käyttäjän näkymässä peräkkäin, mutta niiden ei suinkaan ole välttämätöntä olla peräkkäin kääntäjän näkemässä käännöstiedostossa (.po-tiedostossa) – tai edes sijaita samassa tiedostossa.

Nähdäkseni kysymystä kotoistuksen tekstistä ei voida eikä ole syytä pitää pelkkänä metodisena ongelmana, vaan se on todellinen käsitteellinen haaste ohjelmistokotoistuksessa. Ensimmäinen vaihtoehto ratkaista ongelma on muuttaa tekstin käsitettä niin, että lähtötekstinä ei ole yksittäinen käännösjono eikä edes niiden kokonaisuus, käännöstiedosto, vaan käännösjonot nähdään yhtenä sellaisena resurssina, josta lopullinen teksti muodostuu. Tällöin teksti väkisinkin irtautuisi siitä käsiteperinteestä, jossa se nähdään vain kielellisenä toimintana. Tekstiin pitäisi sisällyttää kaikki se, miten käyttäjä kotoistettavan ympäristön näkee ja kokee – kuvat, värit, liikkeet ja liikeradat, tavat toteuttaa asioita ja niin edelleen.

Otetaan esimerkiksi jokin yksinkertainen ja kaikille tuttu toimenpidesarja: käyttäjän on kopioitava USB-muistitikulta koneelleen tiedosto johonkin kansioon, avattava se asianmukaisella ohjelmalla, tulostettava asiakirja, suljettava tiedosto ja lopuksi poistettava se koneeltaan. Toiminnassa on lukuisia kielellisiä ja sellaisina kotoistettavia elementtejä. Kun käyttäjä esimerkiksi kytkee tikun koneeseen, kone saattaa kysyä, mitä sille tehdään. Käyttäjän pitää avata tikun sisältö johonkin tiedostonhallintaohjelmaan, etsiä tiedosto mahdollisesta kansiorakenteesta, kopioitava tiedosto esimerkiksi pikanäppäimellä (PC-järjestelmissä yleensä Ctrl-C), valitsemalla toimenpide ohjelman kontekstivalikosta (hiiren oikea painike > ”Kopioi”) tai vetämällä ja pudottamalla tiedosto hiiren tai muun osoitinlaitteen avulla kansiosta toiseen. Osa mutta vain osa näistä toimista on kielellisiä ja siten kotoistettavia, mutta käyttäjän näkökulmasta kaikki nämä ovat osa sitä, mitä hänen on tehtävä tavoittaakseen haluamansa lopputuloksen. Siten ne on kaikki mielekästä lukea mukaan samaan ”tekstiin”, koska juuri siitähän kääntämisessä on kyse: merkityksellisen kokonaisuuden ilmaisemista niin, että vastaanottaja sen ymmärtää.

Tekstin käsitteen laajentaminen voi tosin helposti aiheuttaa enemmän ongelmia kuin se ratkaisee. Tekstin suppeahko määritelmä voi olla ymmärrettävämpi ja lähestyttävämpi kotoistamisen kuvauksessa tai opetuksessa. Olisi ehkä mahdollista myös hylätä teksti kotoistamiskääntämisen peruskäsitteenä ja mallintaa käännöstoimintaa teksti(e)n sijaan jonkin muun, esimerkiksi virtuaalisen kielimaiseman käsitteen (Ivković & Lotherington 2009; Ivković 2012) avulla. Oleellista on joka tapauksessa, että tekstiä ei kotoistuksessa nähdä vain kirjoitettuna kielenä vaan loppukäyttäjän toiminnan yhtenä toiminnallisena osana. Kääntäjän toiminnan kohteena puolestaan on tämän toiminnan säätely muun muassa käännösjonoja kääntämällä.

5 Lopuksi

Tässä artikkelissa olen pyrkinyt tuomaan esille, mitä joukkoistettu kotoistaminen on. Ensinnäkin kotoistaminen sinänsä on tärkeä osa globalisoituvaa kulttuuria. Se on kääntämistä, jossa on omat ominaispiirteensä, jotka voivat haastaa perinteisiä tapoja mallintaa kääntämistä. Joukkoistaminen puolestaan on luonteva osa nousevaa vapaiden ohjelmistojen, pilveen sijoitetun tiedon ja verkkoistetun toiminnan kulttuuria eikä sitä pitäisi vierastaa vanhojen epäluulojen vuoksi vaan suhtautua avoimesti uutena mahdollisuutena tuoda kääntämistä ja sen kulttuurista merkitystä tunnetuksi yhä suuremmalle osalle ihmisiä, myös maallikoille. Vaikka joukkoistus alkuun usein hapuilee harrastelijamaisilla poluilla, ammattikääntäjän asiantuntemusta osataan arvostaa ja siitä otetaan kernaasti opiksi.

Lähteet

Bell, Jim & Sharon Loane 2010. ‘New-wave’ global firms. *Journal of Marketing Management* 26:3–4, 213–229.

Berman, Antoine 2007 [1986]. Kääntämisen platonistinen olemus. Käänt. Eetu Viren. Teoksessa: Tapani Kilpeläinen (toim.) *Kääntökirja*. 23° 45. Tampere: Eurooppalaisen filosofian seura, 78–102. [Alk. ilm. Revue d’esthétique 12: 63–73.]

Collis, Betty & Jef Moonen 2008. Web 2.0 tools and processes in higher education: Quality perspectives. *Educational Media International* 45:2, 93–106.

Djelassi, Souad & Isabelle Decoopman 2013. Customers’ participation in product development through crowdsourcing: Issues and implications. *Industrial Marketing Management* 42, 683–692.

Dunne, Keiran J. 2012. The industrialization of translation: Causes, consequences and challenges. *Translation Spaces* 1, 143–168.

Estellés-Arolas, Enrique & Fernando González-Ladrón-de Guevara 2012. Towards an integrated crowdsourcing definition. *Journal of Information Science* 38:2, 189–200.

Fernández Costales, Alberto 2013. Crowdsourcing and collaborative translation: Mass phenomena or silent threat to translation studies? *Hermeneus* 15, 85–110.

Folaron, Deborah A. 2012. Digitalizing translation. *Translation Spaces* 1, 5–31.

FSF = Free Software Foundation 1986. What is free software. Saatavissa: http://fsfe.org/about/basics/freesoftware.en.html [viitattu 30.8.2016]. [Alk. ilm. *GNU’s Bulletin* 1: 1.]

Halliday, M. A. K. 1978. *Language as social semiotic. The social interpretation of language and meaning*. London: Arnold.

House, Juliane 2009. *Translation*. Oxford Introductions to Language Study. Oxford: Oxford University Press.

Ivković, Dejan 2012. Virtual linguistic landscape. A perspective on multilingualism in cyberspace. [Julkaisematon väitöskirja, York University.]

Ivković, Dejan & Heather Lotherington 2009. Multilingualism in cyberspace: Conceptualising the virtual linguistic landscape. *International Journal of Multilingualism* 6:1, 17–36.

Lehtovaara, Aatu 2010. Norsunluutornista basaariin: Nordin tekstianalyysimalli lokalisoinnin tarkastelun pohjana. Pro gradu -tutkielma. Käännöstiede (englanti), kieli- ja käännöstieteiden laitos, Tampereen yliopisto.

Lewis, David, Qun Liu, Leroy Finn, Chris Hokamp, Felix Sasaki & David Filip 2014. Open, web-based internationalization and localization tools. *Translation Spaces* 3, 99–132.

Lievrouw, Leah A. 2010. Social media and the production of knowledge: A return to little science? *Social Epistemology* 24:3, 219–237.

Majchrzak, A. & A. Malhotra 2013. Towards an information systems perspective and research agenda on crowdsourcing for innovation. *Journal of Strategic Information Systems* 22, 257–268.

Mesipuu, Marit 2012. Translation crowdsourcing and user-translator motivation at Facebook and Skype. *Translation Spaces* 1, 33–53.

Munday, Jeremy 2012. New directions in discourse analysis for translation: A study of decision-making in crowdsourced subtitles of Obama's 2012 State of the Union speech. *Language and Intercultural Communication* 12:4, 321–334.

Nieminen, Tommi 2010. *Lajien synty. Tekstilaji kielitieteen semioottisessa metateoriassa*. Jyväskylä Studies in Humanities 136. Jyväskylä: Jyväskylän yliopisto.

Nieminen, Tommi 2011. Retorisia siirtoja ja metodologisia ongelmia crowdsourcing -lokalisoinnin tutkimuksessa. *MikaEL – Kääntämisen ja tulkkauksen tutkimuksen symposiumin verkkojulkaisu* 5, 1–14.
Saatavissa: https://www.sktl.fi/@Bin/85393/Nieminen_MikaEL2011.pdf [viitattu 14.3.2017].

Nogueira, Danilo & Kelli Semolini 2010. Crowdsourcing. *Translation Journal* 14:2.

O'Hagan, Minako 2012. Translation as the new game in the digital era. *Translation Spaces* 1, 123–141.

Olohan, Maeve 2014. Why do you translate? Motivation to volunteer and TED translation. *Translation Studies* 7:1, 17–33.

O'Mahony, Siobhán & Fabrizio Ferraro 2007. The emergence of governance in an open source community. *Academy of Management Journal* 50:5, 1079–1106.

Pettini, Silvia 2015. Auteurism and game localization – revisiting translational approaches: Film quotations in multimedia interactive entertainment. *Translation Spaces* 4:2, 268–288.

Poetz, Marion K. & Martin Schreier 2012. The value of crowdsourcing: Can users really compete with professionals in generating new product ideas? *The Journal of Product Innovation Management* 29:2, 245–256.

Pym, Anthony 2006. What localization models can learn from translation theory. Saatavissa: http://usuaris.tinet.cat/apym/on-line/translation/localization_translation_theory.pdf [viitattu 30.8. 2016].

Stallman, Richard 2009. What means free software and why matters. Saatavissa: http://fsfe.org/ freesoftware/transcripts/rms-2009-05-22-eliberatica.en.html [viitattu 30.8.2016].

Tung, Yuan-Hsin & Shian-Shyong Tseng 2013. A novel approach to collaborative testing in a crowdsourcing environment. *The Journal of Systems and Software* 86, 2143–2153.

Weber, Stefan 2004. *The success of open source*. Harvard University Press, Cambridge (Mass.) & London.

Kirjoittajan esittely ja yhteystiedot

Tommi Nieminen toimii parhaillaan suomen kielen yliopistonlehtorina Itä-Suomen yliopistossa. Sähköpostiosoite: tommi.nieminen (at) uef.fi