Avoimen teknologian ohjelmat, asiakirjat ja rakenteet

Ilpo Kantonen

19.6.2008

Joensuun yliopisto

Tietojenkäsittelytieteen laitos

Pro gradu -tutkielma

Tiivistelmä

Tekijä: Ilpo Kantonen

Tutkielman nimi: Avoimen teknologian ohjelmat, asiakirjat ja rakenteet

Tutkinto: filosofian maisteri

Laitos: tietojenkäsittelytieteen laitos

Oppiaine: tietojenkäsittelyoppi

ACM-luokat: D.2.9

Avainsanat: avoin lähdekoodi, open document, open format, versionhallinta, ylläpito, julkishallinto.

Sivumäärä: 64

Tutkimuksessa selvitetään avoimen lähdekoodin määritelmää, avoimen lähdekoodin ohjelmien tuotantoa prosessina työkaluineen, kotoistuksineen ja julkistuksineen. Tutkimuksessa selvitetään avointen ohjelmien ylläpitoa. Avoimet dokumentit ovat tärkeä osa tietojen käsittelyä. Tutkimuksessa otetaan esille myös muita avoimia rakenteita. Tutkimuksessa selvitetään avointen ohjelmien ja dokumenttien tilannetta julkishallinnossa sekä julkishallinnon avointen lähdekoodien ja dokumenttien projekteja.

Esipuhe

Olen käyttänyt vapaasti jaettavia avoimen ohjelmakoodin ohjelmia kymmeniä vuosia. Tutustuin lähemmin avoimen ohjelmakoodin konseptiin ja niihin ohjelmistoihin asentamalla 14.2.2002 kotitietokoneelleni Linux Red Hat 8.0 -käyttöjärjestelmän. Alkuvaiheiden jälkeen sain järjestelmän toimimaan vakaasti ja huomasin sen toimivan huomattavasti varmemmin kuin Microsoft Windows 98 -järjestelmä.

Syksyllä vuonna 2004 huomasin KDE-työpöytäohjelmiston kcalc-laskinohjelmassa pahan käännösvirheen. Käänteisluvun näppäimessä oli varsin ihmeellinen seliteteksti. Syntyi tarve ja halu korjata tämä. Siitä alkoi tutustuminen KDE:n kielelliseen kääntämiseen ja KDE:n suomennosharrastus. KDE:tä käyttämällä olen tutustunut myös KDE:n virheidenilmoitusjärjestelmään. Aikomus on tutustua myös KDE-ohjelmien koodausta kun aika antaa periksi.

Olen perehtynyt aiemmin LuK-tutkinnon esseen kirjoittamisen yhteydessä ohjelmistojen tuotejulkistukseen ja julkistettujen ohjelmistojen versionhallintaan. Julkistusten hallinta on osa avointen ohjelmistojen elinkaarta. Versionhallinta nivoutuu myös aiheeseen. Tältä pohjalta kiinnostukseni avointen ohjelmistojen aiheeseen on luonnollinen jatkumo.

Avoimien ja yleensä vapaasti jaettavien ohjelmistojen käytön laajeneminen jatkuu. Linuxille on tullut toimistokäyttöön suunnattuja ohjelmia. Julkishallinnolliset virastot ja laitokset sekä elinkeinoelämän yrityksiä on siirtynyt ja siirtymässä Linuxin käyttäjiksi. Linuxin käytöllä ne alkavat samalla käyttää avoimia ohjelmistotuotteita. Linuxille odotetaan lähitulevaisuudessa lisää toimistosovelluksia ja elinkeinoelämän käyttämiä kohdesidonnaisia ohjelmia. Niiden puuttuminen on ollutkin suurin este Linuxin leviämiselle yritysten käyttöön.

Olen avoimen lähdekoodin ohjelmien ja avointen teknologioiden kannattaja.

Lappeenrannassa 19.6.2008 Ilpo Kantonen.

Sanasto

Tässä tutkielman osassa selitetään tutkimuksessa käytettyjä termejä.

Sana	*Selite*
Bugzilla	Virheidenilmoittamisjärjestelmä, jossa on raportointi-ominaisuuksia ja joka osaa ohjata virheet oikeille tekijöille. Bugzillaa käytetään monien avoimien ohjelmistojen kehityksessä.
CCITT	Comité Consultatif International Télégraphique et Téléphonique, International telegraph and Telephone Consultative Committee.
CDF	Compound Document Format. W3C:n (www.w3c.org) standardi, joka kilpailee ODF:n kanssa.
COSS	Centre for Open Source Software. Avointen ohjelmistojen keskus Suomessa.
DMS	Defect Management System. Virheidenilmoitus-järjestelmä. Katso myös VMS.
DRM	Digital Rights Management. Digitaalinen käyttö-oikeuksien hallinta. DRM-tekniikalla rajoitetaan mitä sillä suojatulle tiedolle voi tehdä.
GPL	GPL-lisenssi antaa luvan käyttää ohjelmistoa ja hyödyntää sen koodeja omiin jatkokehitelmiin jopa kaupallisesti.
ISO	International Organization for Standardization
internationalisointi	Ohjelmien kehittäminen sellaiseksi, että ne on helposti muunnettavissa (lokalisoitavissa) erikielisiksi paikallisiin olosuhteisiin sopiviksi.

Sana	*Selite*
Linux	Vapaasti jaettava ja käytettävä ilmainen avoimen lähdekoodin käyttöjärjestelmä.
lisenssi	Käyttöoikeus. Ohjelmaa saa käyttää lisenssin haltija.
lokalisointi	Kotoistus. Ohjelmien tekstien kääntäminen kunkin paikan äidinkielelle, pisteet, pilkut, numeroiden esitysmuodot, valuutat jne.
ODA	Open Document Architecture. ISO:n ja CCITT:n avoimen dokumentin rakenteen määrittely.
open access	Julkisesti saatavilla oleva. Esimerkiksi sivun kirjaston materiaali voi olla open access.
open source	Ohjelman mukana jaetaan vapaasti myös ohjelmakoodi. Ohjelmakoodi on usein GPL-lisenssin alaista, jolloin sitä voi jakaa edelleen ja hyödyntää vaikkapa kaupallisesti.
OSI	Open Source Initiative. Voittoa tavoittelematon organisaatio avoimen lähdekoodin tietouden levittämiseen, käytön edistämiseen, kouluttamiseen ja olla sillanrakentajana eri yhteisöjen välillä.
OSS	Open Source Software. Avoimen lähdekoodin ohjelma tai ohjelmisto.
PDF	Portable Document Format. Adoben kehittämä dokumenttimuoto, joka on suunniteltu dokumenttien siirtoihin ja julkaisutoimintaan.
SDLC	Software Development Life Cycle. Ohjelmisto-kehityksen elinkaari. Tässä tarkoitetaan perinteistä ohjelmiston suunnittelun elinkaarta.
Thesaurus	Oikeinkirjoituksen tarkistus, jossa sanalle etsitään vastaavuussuhteita, hierarkkisia suhteita ja assosiaatio-

Sana	*Selite*
	suhteita muihin termeihin ja sanoihin.
Ubuntu	Yksi Linuxin jakeluversio (distribuutio), joka on saanut paljon huomiota helppokäyttöisyydellään ja käyttäjätuellaan.
VMS	Version Management System. Versionhallintajärjestelmä. Katso myös DMS.
Wikipedia	Avoin tietosanakirja. Kuka vain voi kirjoittaa artikkeleja Wikipediaan. Wikipediassa toimii vertaisarviointi ja tietojen todenperäisyyksiä tarkistetaan.

Sisällysluettelo

1 Johdanto

Tietokoneohjelmat ovat kirjoitettuja toimintaohjeita tietokoneille. Toimintaohjeet käännetään tietokoneen ymmärtämiksi binäärisiksi suorituskelpoisiksi ohjelmiksi tai niitä tulkitaan toiminto kerrallaan toimintaohjeista. Tietokoneohjelmia on ollut olemassa 1950-luvulta lähtien eli siitä lähtien kun tietokoneita on käytetty.

Ohjelmat jaetaan tyypiltään käyttöjärjestelmiin ja sovellusohjelmiin. Käyttöjärjestelmät tuottavat ohjelmallisen rajapintaliittymän sovellusohjelmien ja tietokonelaitteiston välille. Tällöin sovellukset toimivat samanlaisella koodilla laitteistoarkkitehtuurista riippumatta. Ohjelmat ovat olleet yritysten valmistamia 1900-luvulla. Ne on suunniteltu tavallisesti yhteen konearkkitehtuuriin sopiviksi. Ohjelman käyttöoikeudet saa ostamalla ohjelman eikä ilman käyttöoikeuksia ohjelmaa saa juridisesti käyttää.

Tietokoneohjelmia ja niillä käsiteltyjä tietoja on helppo kopioida. Helppo kopioitavuus ovat pohjana ajatukselle vapaista ohjelmistoista. Samanlainen ajatus pätee ohjelmistojen käsittelemille tiedoille. Ensin mainittu filosofia on johtanut vapaiden ohjelmistojen käsitteeseen ja siitä edelleen avoimien lähdekoodien ohjelmistoihin. Jälkimmäinen filosofia on synnyttänyt avointen dokumenttien määritelmän. Käyttöoikeuksiin eli lisensseihin ei tässä tutkimuksessa syvennytä.

Tietokoneohjelmaa käyttääkseen ohjelman käyttäjä tarvitsee vain ohjelman suorituskelpoisen binääritiedoston sekä siihen liittyvät asetus- ja datatiedot. Ohjelmien tuottamisessa lähdekielisiä koodeja ei yleensä jaeta ohjelmien mukana. Kaupallisen ohjelman käyttöoikeus on koskenut perinteisesti vain ohjelman suoritusversiota. Ohjelmakoodi jää ohjelmankehittelijälle. Ohjelman tekijät haluavat säilyttää osaamisensa ja tekniset innovaatiot itsellään. Ohjelman käyttäjä ei ole tavallisesti edes kiinnostunut lähdekoodista, vaan ohjelman funktionaalisesta toimivuudesta ja sopivuudesta hänen tarpeisiinsa. Käyttöoikeuksilla on haluttu myös lisätä ohjelmistovalmistajan tuloja, jotta ohjelmistojen jatkokehittelyyn olisi tarpeeksi resursseja. Käyttöoikeuksilla on pyritty myös kytkemään asiakkaat käyttämään saman tuoteperheen ohjelmia. Ohjelmistojen patentointi ei ole hyvä asia. Se estää uusien innovaatioiden käytön ja edelleen kehittämisen. Patenttikiistat syövät ohjelmistoalan yritysten varoja.

Avoimet ohjelmat tarkoittavat ohjelmia, joiden mukana ohjelman käyttäjälle toimitetaan myös ohjelman lähdekoodit. Avoimiin ohjelmiin luetaan ohjelmat, joiden lähdekoodia käyttäjä voi muokata haluamakseen ja lisätä siihen uusia ominaisuuksia. Public domain -ohjelmilla tarkoitetaan perinteisesti ohjelmia, joiden tekijät luopuvat kaikista oikeuksistaan tehtyyn lähdekoodiin ja sen käyttöön. Nykyiset

kehittyneemmät ohjelmistojen lisenssimäärittelyt ovat korvanneet public domain määrittelyt ja niitä ei enää juuri käytetä. Avoimien lähdekoodien ohjelmien tietoturva on hyvä, sillä koodista voidaan löytää tietoturvaa heikentävät (tahattomat tai tahalliset) kohdat ja niiden mahdolliset sivuvaikutukset.

Avoimet ohjelmistot ovat alkaneet tulla julkisuuteen 2000-luvun alussa suljettujen ohjelmien vaihtoehtona. Avoimien lähdekoodien ohjelmien leviämistä on edesauttanut avoin kehitysympäristö, johon kuka vain voi ottaa osaa. Hyvä esimerkki tällaisesta on Linux. Avoimien ohjelmien käyttö on usein ilmaista, joka säästää niukin resurssein toimivien organisaatioiden ja instituutioiden rahoja ydinliiketoimintaan. Esimerkiksi kunnat käyttivät vuonna 2001 67 miljoonaa euroa pelkästään Microsoftin ohjelmistojen lisenssimaksuihin[1]. Toinen avoimiin ohjelmiin liittyvä käytön motiivi on riippumattomuus ohjelmistotoimittajasta. Tutkimuksessa selvitetään avoimien ohjelmien perusteita ja sovelluskohteita ja ohjelmien ohjelmakehitystä. Tutkimuksen pääosa keskittyy juuri avoimien ohjelmien kehittelyyn, kehitysympäristöihin, tuotantoon, jakeluun erilaisiin käyttöympäristöihin ja niiden tulevaisuutta.

Yksi avoimien ohjelmien käytön leviämiskohde ovat julkinen hallinto ja koulut. Nekin joutuvat elämään niukkenevissa taloudellisissa olosuhteissa. Lisenssimaksuiltaan ilmaiset ohjelmistot ovat tervetullutta. Lisäksi ylläpidon helppous kun järjestelmät on kerran säädetty kuntoon, säästää taloudellisia resursseja.

Internetin palvelimet toimivat pääasiassa Unix- ja Linux-pohjaisina. Niissä avoimet ohjelmat ovat olleet käytössä kautta niiden historian ajan. 2000-luvun alussa Linux alkoi levitä myös työpöytätietokoneisiin.

Julkisuudessa keskustelu avoimien ohjelmistojen ja suljettujen välillä on lähinnä avoimien ohjelmien toimittajien ja Microsoft Corporationin välillä. Microsoft on kasvanut ja päässyt monopolimaiseen asemaan henkilökohtaisten tietokoneiden käyttöjärjestelmävalmistajana. Se on kyennyt vastaamaan ihmisten tarpeisiin koneellisessa tietojenkäsittelyssä. Jo 1990-luvulla monopoliasemaa pidettiin huonoa. Mutta silloin sille ei ollut vielä todellista kilpailijaa.

Seuraavassa luvussa otamme muutaman esimerkin tehdyistä tutkimuksista avoimen lähdekoodin ohjelmista.

1.1 Aiheesta tehdyt tutkimukset

Avointen ohjelmakoodia on tutkittu lähivuosina runsaasti eri näkökulmista ja

1 http://www.digitoday.fi/page.php?page_id=10&news_id=20016523 Microsoftilta jättilasku Suomen kunnille (23.9.2006)

monessa eri tutkimuslaitoksessa. Tutkimuksien suuresta määrästä johtuen otamme tässä tutkimuksessa vain muutaman tutkimuksen ja raportin esimerkiksi.

Jaamme tässä tutkimuksessa tehdyt tutkimukset yleensä avoimiin ohjelmiin liittyviksi tutkimuksiksi, ohjelmistokehitykseen, avoimien koodien ohjelmilla harjoitettavaan liiketoimintaan ja opetukseen liittyviin tutkimuksiin. Tässä yhteydessä otamme mukaan myös Euroopan tasolla tehtyjä tutkimuksia.

Avoimen lähdekoodin kehittämiseen liittyvät tutkimukset:

Vuosittaiset kansainväliset ohjelmistotuotannon konferenssit (ICSE) ovat nähtävillä Internetissä osoitteessa

http://www.icse-conferences.org/

Sivustolla on linkkejä eri vuosien konferensseihin, käsiteltyihin aiheisiin ja sieltä on ladattavissa esitykset PDF-muodossa. Konferensseissa on ollut asiaa myös avoimien lähdekoodien ohjelmista. Esimerkkinä vuonna 2003 ICSE järjesti 3-11. toukokuuta kolmannen työpajan avoimen lähdekoodin ohjelmistotuotannossa. Tämä raportti on ladattavissa PDF-dokumenttina osoitteessa

http://opensource.ucc.ie/icse2003/

Käsiteltyjä aiheita (kaikki aiheet käsittelevät avointa lähdekoodia tai avoimia ohjelmia) ovat olleet ohjelmien markkinat, kehitysprosessi ja työkalut, todisteet kehityksestä, tuotemalli kaupallisessa ympäristössä, politiikan vaikutus, vapaan ohjelmistokehitysympäristön virtuaalinen organisaatiokulttuuri, ohjelmien julkistus, XML-repository sovellusten ylläpitoon ja mukautukseen, ohjelmaprojektien automaattinen mittaaminen, sisällönhallinta ja kehitysportaali, mallinnuksen automatisointi prosesseissa, ohjelmien automaattinen kategorisointi, ohjelmisto-tuotannon mukaanottaminen tietojenkäsittelytieteen opetukseen, integroitu ohjelmistokehityksen aktiviteettiselain, ohjelmistoprojektien laatu ja turvallisuus, kokeellinen arviointi, versionhallinta Linuxin kehityksessä, varmistuksen, oikeellisuuden, avoimen lähdekoodin kehityksen ketteryys.

T. Kainulainen ja J. Valtola ovat kirjoittaneet Kuopion yliopistossa vuosina 2005 – 2006 neljä avoimen lähdekoodin ohjelmien tutkimusprojektin raporttia aiheesta Writing Service using Open Source Application Frameworks.

Vuonna 2003 Toni Strandell kirjoitti Helsingin yliopistossa pro gradu -tutkielman avoimen lähdekoodin tietokantajärjestelmien suorituskyvystä ja skaalautuvuudesta aiheena (Open Source Database Systems: System study, Performance and Scalability.

Avoimen lähdekoodin ohjelmien menestyksestä kirjoitti professori Steven Weber Harwardin yliopistossa vuonna 2004 aiheena The success of open source. (ISBN 0-674-01292-5)

Pearson Education julkaisi Lontoossa vuona 2002 avoimen lähdekoodin ohjelmistotuotannosta Joseph Fellerin ja Brian Fitzgeraldin kirjoittama tutkimus Understanding open source software development. (ISBN 0-201-73496-6)

Fabian Fagerholm on tutkinut Helsingin yliopistosta valmistuneessa pro gradu -tutkimuksessa vuonna 2007 avoimen lähdekoodin ohjelmien projektien laadun mittaamisesta

Nina Helander on kirjoittanut vuonna 2007 Tampereen teknillisessä korkeakoulussa avoimen lähdekoodin ylläpitoympäristöstä tutkimuksessa Open source software management framework. (ISBN 978-951-44-7110-0)

Sanna Hyvönen on kirjoittanut vuonna 2006 opinnäytteen Etelä-Karjalan ammatti-korkeakoulussa aiheena Open source -välineet ja ohjelmistot.

O'Reilly Media on julkaissut Karl Fogelin kirjoittaman kirjan Producing open source software: How to run a successful free software project vuonna 2006. (ISBN 0-596-00759-0.)

Wiley on julkaissut vuonna 2002 Richard Hightowerin ja Nicholas Lesieckin kirjoittaman kirjan Java tools for extreme programming : mastering open source tools including Ant, JUnit, and Cactus. (ISBN 0-471-20708-X.)

Aki Holopainen on Ohjelmistotestauksen open source -työkalut pk-yritykselle. Holopainen Aki. Etelä-Karjalan ammattikorkeakoulu. 2007.

Minna Holopainen on kirjoittanut vuonna 2007 Kuopion yliopistossa väitöskirjan aiheesta Open Source- ohjelmistojen arviointiprosessi pk-yrityksissä.

Timo Koponen on tutkinut Kuopion yliopistossa avoimien ohjelmistojen ylläpitoprosessien arviointia tutkimuksessa tutkimuksessa Evaluation of Maintenance Process in Open Source Software Projects Through Defect and Version Management Systems. (ISBN 978-951-781-988-6, 978-951-27-0107-0 (PDF), ISSN 1459-7586.)

Erika Loimukallio on tutkinut avoimia innovaatioita kandidaatin tutkimuksessaan Open innovation : Open Source Software industry Turun kauppakorkeakoulussa vuonna 2007.

Vuonna 2006 Matti Riikonen on kirjoittanut pro gradu -tutkielman Joensuun yliopistossa aiheena ohjelmistojen lokalisointi ja kansainvälistäminen. Tutkimus selvittää kotoistusta yleisellä tasolla. Tämän tutkimuksen tarkoitus on keskittyä avointen ohjelmien kotoistukseen ja tarkastelunäkökulma on käytännönläheisempi. Päällekkäisyyttä on pyritty välttämään. Tutustuminen Matti Riikosen tekemään tutkielmaan antaa hyvän lisän tutkimukseni kotoistusosaan.

Käyttöoikeudet ja lisensointi:

Mikko Mustonen on kirjoittanut vuonna 2003 Helsingin yliopistossa tutkielman

Copyleft - the economics of Linux and other open source software software.

Avoimilla ohjelmilla harjoitettavaan liiketoimintaan liittyvät tutkimukset:

Tampereen yliopiston hypermedialaboratorio on julkaissut vuonna 2007 Nina Helanderin, Timo Aaltosen, Teemu Mikkosen, Ville Oksasen, Mikko Puhakan, Marko Seppäsen, Tere Vadénin ja Niklas Vainion tutkimuksen aiheesta Open Source Software Management Framework.

Nina Helander ja Maria Mäntymäki ovat kirjoittaneet tutkimuksen Empirical insights on open source software business Tampereen teknillisessä korkeakoulussa vuonna 2006. (ISBN 951-44-6669-1)

Oulun ammattikorkeakoulussa on valmistunut vuonna 2006 opinnäyte aiheena Open source -pohjainen projektinhallinta. Sen on kirjoittanut Lea Vesikukka. Vertailussa tutkitaan OpenWorkBench, GanttProject, Gnome Planner for Windows ja AgileTrack -ohjelmistoja.

Vuona 2007 Pia Satopää on pitänyt Turun ammattikorkeakoulussa seminaarin aiheena Open source - avoin lähdekoodi liiketoiminnan osana.

Etla on julkaissut vuonna 2006 Arto Sepän kirjoittaman tutkimuksen aiheena Open source in finnish software companies.

Ohjelmien suojauksesta lisensseillä ovat tutkineet ja kirjoittaneet kirjan Ruben van Wendel de Joode, Hans de Bruijn ja Michel van Eeten. Kirjan on julkaissut T.M.C.Asser Press on julkaissut sen vuonna 2003 (ISBN 90-6704-159-9). Kirjan nimi on Protecting the Virtual Commons: Self organizing open source and free software communities and innovative intellectual property regimes.

Opetuksessa:

Avoimella lähdekoodilla on tehty opetusympäristöön Moodle kurssienpitojärjestelmä, joka toimii Internetin kautta. Tämän ohjelmiston käyttöön kurssien suunnilttelussa on selvitetty tutkimuksessa Moodle. E-learning course development. Rice, William H, IV. Birmingham, Packt 2006. ISBN 1-904811-29-9.

Hallinnollisella tasolla Suomessa ja Euroopassa:

Pekka Peltola on tutkinut avoimen lähdekoodin käyttöä julkisyhteisöjen tietojärjestelmissä vuonna 2008 valmistuneessa pro gradu tutkielmassaan Tampereen yliopistossa.

EU rahoittaa laajan FLOSSWorld-tutkimuksen avoimen koodin vaikutuksista Euroopassa. Tutkimuksen tarkoitus on selvittää avoimen koodin vaikutusta teknisten

taitojen kehittymisessä, taloudelliset vaikutukset ja työpaikkojen luonti, alueelliset erot ohjelmakehityksessä sekä julkishallinnon mahdollisuudet käyttää avointa koodia hyväkseen[2].

1.2 Tutkimuksen tavoitteet

Tutkimuksen tarkoituksena on selvittää avoimia teknologioita. Tässä yhteydessä teknologioilla tarkoitetaan avoimia ohjelmia ja avoimia dokumentteja. Tutkimuksessa painotetaan avoimien ohjelmien ohjelmistokehitystä, kehitysympäristöjä, kotoistusta ja tuloa markkinoille olemassaolevien ratkaisujen vaihtoehdoksi.

Tutkimuksessa perehdytään ohjelmankehitykseen avoimena kehitysympäristönä, ohjelmistotuotantoprosessina, käytettyjä työkaluja ja kotoistusta (lokalisointia). Kotoistuksessa tarkastellaan lähinnä Linux-järjestelmän KDE-käyttöliittymä-ohjelmistoa. Tavoitteena on verrata niitä suljettuihin ohjelmiin ja niiden kehittely-nopeutta varsinkin virheenkorjauksissa. Tutkimuksessa selvitetään myös avointen ohjelmien tuotejulkistusta ja jakelua. Tämä osa on jatkoa vuonna 2004 kirjoittamalleni tuoteversionhallinta -esseelle. Avoimista dokumenteista (open document) selvitetään niiden standardia ja rakennetta. Standardia verrataan suljettuihin standardeihin ja pohditaan avoimuuden etuja.

Tutkimuksessa pohditaan, onko avoimien ohjelmien ohjelmakehitys laadukkaampaa suljettuihin ohjelmiin verrattuna. Avoimien ohjelmien kehittäjien sitoutuminen kehitystyöhön vaihtelee paljon enemmän kuin kaupallisissa ohjelmissa, joiden kehitystä tahdittavat yritysten tulosvaatimukset.

Tutkimuksessa selvitetään avointen ohjelmien kotoistusta. Teoreettinen pohdinta on jätetty pois, sillä se on valmiina Matti Riikosen kirjoittamassa aihetta käsittelevässä pro gradu -tutkielmassa vuonna 2006. Tutkimus keskittyy kotoistukseen avointen ohjelmien kehitysympäristössä.

Tutkimuksessa selvitetään avoimien ohjelmakoodien ja teknologioita julkis-hallinnollisissa kohteissa. Viranomaisten valmiutta avoimiin dokumentteihin selvitetään. Avoimien teknologioita selvitetään myös koulumaailmassa.

2 http://www.digitoday.fi/page.php?page_id=9&news_id=200511941 Digitoday: EU rahoittaa suuren avoimen koodin tutkimuksen (9.3.2007)

1.3 Teoriakehys

Avoimien lähdekoodien ohjelmien tutkimus on tietojenkäsittelytieteen osa. Se sivuaa myös sosiologiaa avoimissa kehitysympäristöissä, psykologiaa kehitystyöhön sitoutumisessa ja vapauden tunteessa, taloustieteitä ohjelmistojen tuotannon kannattavuuden ja tehokkuuden sekä käytön kannattavuudessa. Lakitiedettä sivutaan lisensseissä ja ohjelmistopatenteissa. Avoimien ohjelmien kotoistus sivuaa kansantiedettä ja kulttuuria. Jopa ekologiaa sivutaan kun ajatellaan ohjelmia uusiutuvana luonnonvarana[3].

Tutkimus on avoimien teknologioiden ja niiden teorioiden esittely kirjallisuuden perusteella. Kirjallisuuden lisäksi on runsaasti Internetistä löytyviä lähteitä. Tutkimuksessa on mukana paljon kirjoittajan omaa käytännön kokemusta varsinkin muutamasta avoimen ohjelmiston kotoistuksesta.

1.4 Tutkimuksen rakenne

Tutkimus jakautuu johdanto-osan jälkeen kahdeksaan lukuun. Luvussa 1 eli tässä johdanto-osassa perehdytään olemassaoleviin tutkimuksiin avointen lähdekoodien ohjelmista. Johdanto-osassa asetetaan tavoitteet tälle tutkimukselle. Tieteellinen viitekehys selvitetään. Lopuksi johdanto-osassa esitellään tutkimuksen rakenne.

Luvussa 2 tarkastellaan avoimen lähdekoodin ohjelmien perusteita. Tutkimme niitä yleensä ja motiiveja tehdä avointa koodia sekä avoimuuden mahdollistamaa yhteisöllisyyttä. Avoimen lähdekoodin ohjelma verrataan suljetun lähdekoodin ohjelmiin. Avoimen lähdekoodin mahdollisuuksia tarkastellaan maapallon (köyhillä) alueilla, joilla tietokoneita ei ole aiemmin ollut yleisessä käytössä.

Avoimet dokumentit (ODF) ovat yhtä tärkeä osa tietojen käsittelyä kuin avoimet ohjelmat. Avoimia dokumentteja käsitellään luvussa 3. Tarkastelussa otetaan esille avoimien dokumenttien standardisointi ja lisäksi niitä tukevat avoimen lähdekoodin sekä suljetun lähdekoodin ohjelmat.

Avoimia dokumentteja aiemmin on kehitelty avoimia teknologioita mm. äänentoiston ja videokuvan käsittelyyn. Näitä muita avoimia teknologioita tarkastellaan luvussa 4.

Avointen lähdekoodien ohjelmistojen ohjelmistotuotantoa ja ohjelmistotuotannon ympäristöä käsitellään tutkimuksen luvussa 5. Ohjelmistotuotantoon kuuluvat ohjelmankehitysympäristöt, työkalut, versionhallinta ja kotoistus sekä julkistus.

Ohjelmistojen tuotannon jälkeen ohjelmien elinkaareen kuuluu käyttö ja ylläpito. Ylläpitoa tarvitaan ohjelmien sujuvan käytön varmistamiseksi. Avoimien ohjelmistojen ylläpitoa käsitellään luvussa 6.

---

3 The Grill, Eben Moglen, ComputerWorld Janyary 21, 2008

Luvussa 7 tutustutaan kolmeen avoimen lähdekoodin sovellukseen ja kahteen avoimen lähdekoodin ympäristöön.

Luvussa 8 tarkastellaan avoimien teknologioita julkisessa hallinnossa niiden leviämisen kannalta. ODF on merkittävässä roolissa. Koulumaailman teknologioita käsitellään erikseen ja sitten lopuksi muita julkisia palveluja.

Luvussa 9 arvioidaan tutkimusta ja kootaan yhteenveto.

2 Avoimen lähdekoodin ohjelmat

Avoimen lähdekoodin ohjelmat ovat syntyneet vapaiden ohjelmien pohjalle. Seuraavassa selvitämme, mistä vapaat ohjelmistot ovat saaneet alkunsa ja miten avoimen lähdekoodin ohjelmat liittyvät niihin.

Avoimien lähdekoodien ohjelmien jakeluun liitetään lisenssejä. Näitä lisenssejä ovat Public Domain, Shareware, Freeware, GNU General Public License (GPL), GNU Lesser General Public License (LGPL), Mozilla Public License (MPL), BSD-lisenssi. Tässä tutkimuksessa emme perehdy erilaisiin lisensseihin. Otamme tarkasteluun myöhemmin GPL-lisenssin tarvittavin osin.

Ohjelmien kehittäjät pitävät ohjelmistopatentteja huonona asiana. Ohjelmistopatentti estää edistyksellisten menetelmien käytön laajenemisen ohjelmissa. Tästä puhumme enemmän omassa luvussaan.

Avoin koodi on ollut alkuvaiheessaan vaihtoehto suljetulle koodille. Avoimen koodin alkuvaiheen leviämisvaiheen jälkeen avoimuus ei tule olemaan samanlainen toisensa poissulkeva vaihtoehto. Esimerkiksi Microsoft on saanut avoimen koodin auktoriteettiasemassa olevalta OSI:lta Microsoft Public License (Ms-PL) ja Microsoft Reciprocal (Ms-RL) lisenssin. Tämä on mahdollista kun Microsoftin lisnessi täyttää OSI:n määrittelemät 10 ehtoa. Silloin niitä voidaan markkinoida OSI:n hyväksymänä avoimena koodina.

Suljetun ja avoimen ohjelmakoodin kehittäminen ei ole aina vastakkain asettelua. Suljetun koodin kehittäjä voi tukea avoimen ohjelmakoodin kehitystä[4]. On tietenkin tehtävä kysymys, onko kaupallisesti voittoa tavoittelevalla yhtiöllä omia intressejä tässä avustamisessa. Tässä yhteistyössä esimerkiksi Microsoftilla voi olla tarkoituksena antaa tietoturvaltaan parempi selain heidän käyttöjärjestelmiinsä. Tai sitten Microsoft näkee, että FireFox tulee olemaan kilpaileva tuote Internet Explorerille. Ilman yhteistyötä se tulisi syrjäyttämään Internet Explorerin. Koska Internet Explorer on kiinteästi sidottu Microsoft Windows -käyttöjärjestelmään, on vaara, että käyttäjät siirtyvät käyttämään muita käyttöjärjestelmiä ilman yhteistyötä.

Vapaiden avoimien ohjelmistojen käyttäjien (asennusten) lukumäärää on vaikea arvioida. Lisenssillisten ja maksullisten ohjelmien käyttäjien määrä näkyy ostettujen tuotteiden lukumääränä. Mahdollinen arvioitu piratismi voidaan lisätä tähän lukuun. Rekisteröityjen ohjelmistotuotteiden käyttäjien lukumäärä on myös helppo saada selville. Vapaiden avointen ohjelmien käyttö on vapaata ilman pakollista

4 http://www.tietokone.fi/uutta/uutinen.asp?news_id=27784 DigiToday: Microsoft auttamaan Firefoxin kehitystyössä (2.10.2006)

rekisteröintiä. Näin ollen käyttäjien lukumäärän arviointi on ohjelmapakettien latausten lukumäärän varassa. The Open Source Census käynnisti huhtikuussa 2008 laajan kartoitusprojektin avoimen lähdekoodin ohjelmien käytön arvioimiseksi. (OSSCENSUS, 2008)

2.1 Vapaa ohjelmisto

Vapaan ohjelmiston filosofia lähtee ajatuksesta, että hyvää on voitava jakaa. Jos joku omistaa hyvän ohjelman, hän haluaa jakaa sen toisille. Myös toiset haluavat hyvän ohjelman käyttöönsä. Tästä ajatuksesta lähti vuonna 1983 Richard Stallmanin perustamat GNU-projekti ja Free Software Foundation -niminen voittoa tavoittelematon järjestö, jonka tarkoituksena on valistaa vapaiden ohjelmien eduista. (GNU, 1984) (FSF, 1985)

GNU-projektin filosofian mukaan vapaata ohjelmistoa saa suorittaa mihin tahansa tarkoitukseen. Ohjelman toiminnallisuutta saa tutkia ja soveltaa sitä omiin tarpeisiin. Vapaata ohjelmistoa saa jakaa kopioita edelleen. Vapaata ohjelmaa on vapaus parantaa ja julkistaa omat parannukset avoimesti, jotta muutkin voisivat hyötyä kehitystyöstä. Jakamista voidaan rajoittaa ns. copyleft-säännöllä, joka estää jakajan lisäämästä omia rajoituksia vapaalle ohjelmalle.

Vapaita ja avoimen lähdekoodin ohjelmia pidetään usein synonyyminä ilmaisille ohjelmistoille. Tämä voi johtua tutkielman kirjoittajan mielestä siitä, että avoimen lähdekoodin ohjelmia ajatellaan vaihtoehtona kaupallisille ohjelmille. Erityisesti tämä tulee esille monopolimaisen aseman omaavien ohjelmistojen valmistajien takia. He ovat esimerkiksi voineet luoda ohjelmistoistaan sellaisia, että ne toimivat parhaiten vain heidän toisten kaupallisten ohjelmien kanssa.

Vapaita ohjelmistoja koskevat säännöt eivät kuitenkaan koske kaikkia avoimen lähdekoodin lisenssejä – aina ei vaadita, että myös muokatut versiot annettaisiin vapaaseen käyttöön. GNU-projektin ja Free Software Foundationin johtaja Richard Stallman arvosteli avoimen lähdekoodin liikettä pragmaattisuudesta ja vapaiden ja "puolivapaiden" ohjelmien rajan hämärtämisestä. (Peter Larsson, 2008) Ristiriidat eivät ole kuitenkaan olleet yhteensovittamattomia, vaikka peruslähtökohdissa on eroja.

Vapaiden ohjelmistojen yhtenä heikkoutena on tuotetuen puuttuminen. Tuotetuki on vapaaehtoisuuden varassa. Harrastelijakäyttäjällä voi käyttää aikaa ratkaisujen etsimiseen ja hän voi jättää käyttämättä ohjelmia, kunnes ratkaisu on löytynyt. Liiketaloudellisessa toiminnassa sellainen ei ole mahdollista. Tuotetuella on siksi suuri merkitys, eikä tuotetuen maksullisuus ole estävänä kriteerinä käytölle.

Avoimien lähdekoodien Linux-käyttöjärjestelmissä esimerkiksi SuSe on tunnettu Novellin tukemana ja näin ollen yrityskäyttöön sopivana.

Avoimien ohjelmakoodien ohjelmien kehittämisessä kehitystyötä tehdään yleensä tarpeen tai harrastuksen mukaan. Kehitystyö ei ole projektiluonteista tiukkoihin aikatauluihin sidottua ohjelmistotuotantoa. Käyttäjiltä tulleita tarpeita ovat mm. ohjelman käyttäjiltä tulleet virheraportit. Käyttäjiltä tulleet sekä ohjelman tekijän omat kehitysideat toteutetaan löysällä aikataulutuksella.

On mielenkiintoista pohtia, miksi avoimien ohjelmakoodien ohjelmat, avoimet dokumentit ja yleensä tietojenkäsittelyn avoimuus on tullut ajankohtaiseksi juuri 2000-luvun taitteessa. Onko syynä sattuma vai luonnollinen kehitysvaihe ohjelmistotuotannossa sekä tietokoneiden käytössä? Vai onko tietojenkäsittelyn kannalta ulkoiset tekijät aiheuttamassa siirtymistä avoimuuteen? Edelleen voidaan pohtia, voiko tietojenkäsittely tulla kypsäksi kehittyessään.

Käsitysteni mukaan tietoteknistä tietotaitoa on ollut aiemmin vähän ja se on ollut harvojen käsissä. Kysyntää tietotaidolle on kuitenkin riittänyt. Silloin markkinoilla on pätenyt ajatus: ”Meillä on tietotaitoa, osta sitä meiltä.” Nyt kun ohjelmistokehitys on saavuttanut kyllästymispisteen, sitä on tarjolla kilpailuun asti. On syntynyt asiakkaan markkinat. Onko tämä kehitys synnyttänyt maaperää avointen ohjelmistojen käytön laajenemiselle? Avoimien lähdekoodien ohjelmia on ollut saatavilla kautta aikojen saatavana, mutta nyt niiden käyttö ja kehitys on laajentunut huomattavasti.

Kilpailun ylläpitäminen on käyttäjien kannalta yleinen etu. Ohjelmistojen tuottajat parantavat tuotteitaan ollakseen kilpailijoitaan parempia. Käyttäjän kannalta ohjelmatuotteet paranevat itsestään. Tästä kehityksestä hyötyvät muutkin tahot. Kaupankäynti vilkastuu.

Kilpailua edistävät myös avoimet dokumentit. Ne mahdollistavat halutun sovellusohjelman käytön ja vaihdon kivuitta kun käsiteltävä tieto on avoimessa muodossa. Avoimet dokumentit ovat yhtä tärkeä asia kuin avoimien lähdekoodien ohjelmat. Nehän ovat tietoa, jota avoimet ohjelmistot käsittelevät. Tieto, jota käsitellään on yhtä tärkeää kuin sen käsittely. Rajan selvyys XML-dokumenteissa tietojen ja niiden käsittelyn erillisyyteen vähenee. On epäselvää, onko se tietoa vai sovellusta.[5]

5 Are Documents Apps? Computerworld January 28, 2008.

2.2 Avoin lähdekoodi

Open Source Initiative (OSI) on Bruce Perensin ja Eric S. Raymondin helmikuussa 1998 perustama organisaatio, jonka tarkoitus on edistää avoimen lähdekoodin ohjelmistojen käyttöä. Open Source Initiative on julkaissut avoimen lähdekoodin ohjelmistoille kymmenkohtaisen määritelmän. (OSI, 1998) Ehdot täyttävä ohjelmisto luokitellaan avoimen lähdekoodin ohjelmistoksi. OSI:n määritelmä muistuttaa vapaan ohjelmiston määritelmää, mutta on osittain löyhempi. Kaikki vapaat ohjelmistot ovat myös avoimen lähdekoodin ohjelmistoja, mutta kaikki avoimen lähdekoodin ohjelmistot eivät ole vapaita.

Yksi avoimen lähdekoodin määritelmä löytyy The Open Source Definition -sivulta[6]. Määritelmässä on huomionarvoista, että avoimen lähdekoodin määritelmä ei ole sinänsä lisenssi. Se on määritelmä luvallisista ohjelmalisenssissä ollakseen avoimen lähdekoodin ohjelma.

Avoin lähdekoodi (engl. open source) tarkoittaa mitä tahansa ohjelmistoa, jonka lisenssi täyttää Open Source Initiativen (OSI, 1998) määrittelemät vaatimukset. Avoimen lähdekoodin määritelmän pääkohdat ovat seuraavat:

- Ohjelman täytyy olla vapaasti levitettävissä ja välitettävissä.
- Lähdekoodin täytyy tulla ohjelman mukana tai olla vapaasti saatavissa.
- Myös johdettujen teosten luominen ja levitys pitää sallia.
- Lähdekoodin suora muokkaaminen voidaan kuitenkin kieltää, jolloin muutokset ja korjaukset on toimitettava erillisinä lisäyksinä. Voidaan myös vaatia, ettei johdettua teosta levitetä samalla nimellä tai versionumerolla kuin lähtöteosta.
- Yksilöitä tai ihmisryhmiä ei saa asettaa eriarvoiseen asemaan.
- Käyttötarkoituksia ei saa rajoittaa.
- Kaikilla ohjelman käsiinsä saaneilla on samat oikeudet.

Avoimen lähdekoodin ohjelmien koodissa lisenssi ei saa olla riippuvainen laajemmasta ohjelmistokokonaisuudesta, jonka osana ohjelmaa levitetään, vaan ohjelmaan liittyvät oikeudet säilyvät, vaikka se irrotettaisiin kokonaisuudesta.

6 http://perens.com/Articles/OSD.html (2.10.2006)

Avoimen lähdekoodin lisenssi ei voi asettaa ehtoja muille ohjelmille. Ohjelmaa saa levittää myös yhdessä sellaisten ohjelmien kanssa, joiden lähdekoodi ei ole avointa.

Lisenssin sisällön pitää olla riippumaton teknisestä toteutuksesta. Oikeuksiin ei saa liittää varauksia jakelutavan tai käyttöliittymän varjolla.

Tekijänoikeudet säilyvät koodin kirjoittajilla. OSI (Open Source Initiative) on tarkastanut ja hyväksynyt joukon lisenssejä, joiden mukaisia ohjelmistoja pidetään avoimen lähdekoodin ohjelmistoina. Niihin kuuluvat perinteisten GNU GPL:n ja BSD-lisenssien lisäksi myös eräiden yhtiöiden omat lisenssit, joilla ne jakavat ohjelmistojaan, kuten Apache-ryhmän, IBM:n (IBM Public License), Intelin (Intel Open Source License), Applen (Apple Public Source License), Nokian (Nokia Open Source License), Sun Microsystemsin (Sun Industry Standards Source License ja Sun Public License) ja Mozillan (MPL 1.0 ja 1.1) lisenssit.

OSI ei ole saanut varattua itselleen tavaramerkkiä "Open Source", mutta se on saanut merkittävästi vaikutusvaltaa yritysmaailmassa ja onnistunut rajoittamaan termin käyttöä.

Kaupallisten ohjelmien tuotannossa ohjelmia tuottavilla yrityksillä on tavoitteena voiton maksimointi. Voiton maksimoinnilla voi olla vaikutusta siihen, mitä ominaisuuksia ohjelmiin kehitetään ja osia niistä kehitetään. Yritykset haluavat kehittää niitä ominaisuuksia, joita ostajat haluavat. Ohjelmista tehdään visuaalisesti näyttäviä uusia ominaisuuksia. Tämä taas voi syrjäyttää virheiden korjaamiseen käytettyjä resursseja (kunnes käyttäjät vaativat tarpeeksi ponnekkaasti korjauksia). Tämä kehitys on nähtävänä kaupallisissa ohjelmissa.

Valinnan vapaus on arvo, jota avoimen lähdekoodien ohjelmat ja avoimet dokumentit toteuttavat. Avoimissa ohjelmissa markkinoilla on useita saman tehtävän tekeviä ohjelmia. Käyttäjä voi valita mieleisensä niiden joukosta. Onko vaihtoehtoisia ohjelmia runsaasti tarjolla? Esimerkiksi KDE ja Gnome -järjestelmissä ohjelmat toimivat ristikkäin, sillä kumpikin käyttää samaa GTK-grafiikkakirjastoa. Tästä syystä käyttäjän on helppo vaihtaa ohjelmaa, kunhan ohjelman tiedot (data) voidaan muuntaa toiselle ohjelmalle sopivaksi. Avoimet dokumentit luovat teoriassa pohjan täydelliselle yhteensopivuudelle, jolloin tietojen epäyhteensopivuutta ei tietojen rakenteen vuoksi ole.

Ohjelmien hankinnassa ohjelmista otetaan selvää esitteiden kautta, kysellään käyttökokemuksia ja mahdollisesti kokeillaan kokeiluversiota omassa järjestelmässä. Valinnan jälkeen sitoudutaan tehtyyn ratkaisuun. Sitoutumisesta voi olla haittapuolia esimerkiksi laitteistovaatimusten takia. Tietokoneiden laitteistoihin joudutaan hankkimaan kalliita lisäominaisuuksia tai tehon lisäystä. On eduksi, jos sellaisessa

vaiheessa voidaan siirtyä toisen ohjelman käyttäjäksi. Toisen ohjelman käyttämät menetelmät ja resurssivaatimukset voivat olla erilaisia ratkaisuista riippuen. Näin ollen voidaan säästyä kalliista laitteistoinvestoinneista.

Kilpailutus on päivän sana 2000-luvulla julkisissa hankinnoissa. Kaupallisella puolella se on ollut jo aiemmin arkipäivää. Kilpailutus on on helpompaa ja suo laajemmat vaihtoehtovalikoimat kun järjestelmät ovat yhteensopivia. Avoimet koodit ja standardit luovat pohjan yhteensopivuuden parantumiselle.

2.3 Motiiveja

Avoimen lähdekoodin ohjelmistojen tuottamisen motiivit ovat erityisen kiinnostavia. Miksi kukaan tekisi ohjelmia, joista ei hyödy kun tulokset ovat kaikkien nähtävillä ja hyödynnettävissä? Vai hyötyykö? Aihetta tutkineet Feller ja Fizgerald (Feller & Fitzgerald 2002) jaottelevat motivaatiot kahteen pääluokkaan mikro- ja makrotaso. Näillä tasoilla motiiveja voidaan jaotella teknologiseen, taloudelliseen ja yhteiskunnalliseen motivaatioon.

Merkittävä motivaatio kehittäjillä on halu kehittyä ohjelmien tuottajina oppimalla toisilta. Toisten auttamishalu on myös merkittävä ominaisuus avoimien ohjelmakoodin ohjelmilla. Mottona on: kun minä annan ja joku toinen ja kolmas antaa, me kaikki hyödymme kumulatiivisesti. Kumulatiivisuuteen liittyy myös lisäarvo, joka syntyy yhdistämisessä. Avointen ohjelmien kehittämisen tietotaitoa siirretään myös kaupallisesti hyödynnettäviin ohjelmistotuotteisiin.

Motiivina avoimen lähdekoodin ohjelmien kehitykselle voi olla vapaa-ajan järkevä käyttö. Kuten on todettu, vapaiden ohjelmien kehittäjät saavat ansiotulonsa muusta työstä ja tekevät avointa koodia sen ohella.

2.4 Yhteisöllisyys

Avoimissa ohjelmistoissa yhteisöllisyys näkyy sekä ohjelmien käytössä, että kehittelyssä. Ohjelmien käyttö rohkaisee käyttäjiä tukemaan toinen toisiaan (free user to user assistance). Tätä voi verrata vertaistukeen. Tällaista vertaistukea annetaan suljettujen koodien ja kaupallisten ohjelmien käyttäjien kesken. Kaupallisten ohjelmien käyttökustannuslaskelmissa ei oteta mukaan tällaista vertaistukea, sillä sitä on vaikea mitata.

Myös kehityksessä yhteisöllisyys on näkyvissä. Koodin kehittäjät julkaisevat kehitysversioita julkisina ja saavat jo ennen varsinaista julkistusta ilmoituksia

ohjelmavirheistä ja kehitysehdotuksia. Parhaimmassa tapauksessa he saavat valmiin paikkakoodin ohjelmaan (silloin ei tarvitse tehdä itse mitään). Myös suljetun koodin ohjelmakehitys saa palautetta ennen varsinaista julkistusta testauksen alfa-, beta- ja gammaversioiden julkistamisella. Palaute keskittyy kuitenkin vain näihin julkistuksiin eikä ole jatkuvaa kuten avoimilla ohjelmilla.

Avoimessa kehitysympäristössä ohjelman kehittäjiä on paljon ja kuka tahansa voi liittyä mukaan. Tällainen luo pohjan ryhmän tekemän työn lisäarvolle. Lisäarvo tulee siitä kun eri kehittäjät oppivat toisiltaan uusia menetelmiä ja uusia heille tuntemattomia ideoita. Tämä lisäarvo on kumulatiivista.

Avointa koodia kehitetään joko vapaaehtoisin voimin tai yrityksen palkkaamana. Tutkimusten mukaan on huomattu, että jotta ohjelmistokehitys onnistuisi, täytyy kahden asian olla kunnossa: 1) tekijällä on turvattu toimeentulo eli rahaa elää ja 2) tekijällä riittää tarpeeksi vapaata aikaa koodata ohjelmistoa.

Avointen ohjelmistojen kehittäjiä löytyy Euroopasta ja Amerikasta. Mutta esimerkiksi Intiassa heitä ei löydy niinkään paljoa. Se johtuu siitä, että turvatun toimeentulon ehto täytyy harvassa tapauksessa. Avoimet kenen tahansa osallistuttavissa olevat projektit mahdollistavat myös avoimuuden kehitystä haittaavia negatiivisia lopputuloksia. Esimerkiksi Wikipedian avoimuuden tulevaisuus on viime aikoina lisääntyneen häiriköinnin takia asetettu kyseenalaiseksi[7].

Avoimet kehitysympäristöjen hajanaisuus merkitsee myös pienempää sitoutuneisuutta ohjelmien kehitystyöhön. Avoimen koodin ohjelman kehittäjän elämäntilanne voi suosia kehittämistyötä projektin aloitushetkellä. Myöhemmin tilanne muuttuu. Projekti saattaa jäädä unohduksiin. Tutkielman kirjoittaja asensi tietokoneellensa avoimen lähdekoodin peliohjelman Atlantik (KDE työpöytäohjelmassa), jota hän oli kokeillut noin kaksi vuotta aiemmin. Peli oli kehittynyt huomattavasti sinä aikana. Kävi kuitenkin ilmi, että tekijä ei enää kehitä tätä peliä ja kehitysversioiden lähdekoodit ovat mahdollisesti kadonneet! Niitä ei ole saatavilla julkisilla ohjelmankehityspalvelimilla. Näin tapahtui siitä huolimatta, että uusista kehitysversioista oli tietoa www-sivuilla kuvaruudunkaappauskuvineen.

2.5 Vastarinta avoimelle koodille

Avoimen koodin määritelmän rinnalle on syntynyt puoliavoimen koodin määritelmä. Se sallii koodin näkemisen, mutta muutoksia siihen ei saa tehdä. Microsoft

7 http://www.coss.fi/fi/ajankohtaista/wikipedian-tulevaisuus-puhuttaa-openmind-konferenssissa.html Wikipedian tulevaisuus puhuttaa Openmind-konferenssissa (7.10.2006)

corporation on luonut uuden avoimen lähdekoodin johdannaisen ”Shared Source”. Se on avoin koodi, mutta sitä ei saa muuttaa eikä jakaa edelleen. Tästä johtuen avoimen lähdekoodin ohjelmiin ei voi ottaa mukaan mitään Shared Source -koodia (mukaan liittäminen rikkoisi tekijänoikeuksia). Tästä on seurauksena se, että avoimen lähdekoodin ohjelmissa ei saa käyttää samaa koodia. Samankaltainen idea on HP:n ”Corporate Sourcessa”, jota voidaan käyttää ainoastaan HP-yhteisössä. (Feller & Fitzgerald, 2001) Tällainen voi rajoittaa avoimen lähdekoodin kehitystyötä. On esitetty epäilyjä, että sellaista on tehty jopa tarkoituksella

Käyttäjän kannalta siirtyminen avoimiin ohjelmiin on muutos. IT-alalla on tullut tutuksi sanonta ”vain muutos on pysyvää”. Millaista muutosvastarintaa avoimet ohjelmistot voivat kohdata? Muutosvastarintaa voi syntyä psykologisesti siitä, että vanhoista toimintamalleista on luovuttava. Uuden opettelu merkitsee työtä. Esimerkiksi ohjelmassa käytetyt tavat kopioida ja liittää tekstiä ovat juurtuneet syvälle käyttäjän alitajuntaan. Uuden (avoimen) ohjelman erilainen tapa tuottaa aluksi virheellisiä toimintoja kun käyttäjä tekee asiat ulkomuistista kuten ennenkin. Seuraavassa vaiheessa hän muistaa toimia uudella tavalla, mutta sen työstäminen vie ajattelutyötä ja hidastaa varsinaisen tehtävän suorittamista. Vasta pitkän ajan kuluttua hän omaksuu uuden tavan alitajuntaan ja työskentely koneella sujuu kuten aiemminkin.

Ohjelmat voivat olla myös erinäköisiä. Joitakin tuttuja toiminnallisuuksia ei ole ja on liuta uusia. Jotkut toiminnot voivat olla eri valikoissa. Käyttäjä ei niitä löydä. Tunnettua on, että ohjekirjaan tartutaan vasta viimeisessä hädässä kun muu ei enää auta.

Avoimet ohjelmat voivat synnyttää taloudellista muutosvastarintaa. Suljettujen lähdekoodien ohjelmistojen tuottajat menettävät rahanlähteitään kun heidän ohjelmistojen käyttö vähenee ja sen myötä ohjelmistotuotteita menee vähemmän kaupaksi. Luonnollisesti voittoa tavoitteleva yritys pyrkii varmistamaan tuotteiden menekin.

Tällaisesta kaupallisen tuotteen myynnin edistämisestä on löydettävissä esimerkkejä. Yritys voi tehdä tutkimuksia, jotka osoittavat heidän tuotteen ylivertaisuuden. Niissä osoitetaan ylläpitokustannukset huokeiksi, ohjelmiston ominaisuudet paremmiksi. Myös lisensiointia on käytetty tarkoituksena estää vapaiden avointen lähdekoodien ohjelmien leviämistä markkinoille.

2.6 Suljetun ohjelmistokoodin epäkohtia

Suljetun koodin ohjelmien pahin epäkohta on se, että ohjelman sisäistä toimintaa ei

tiedetä. Se on kuin musta laatikko, jonne syötetään lähtötiedot ja saadaan tulostiedot. Virheiden korjausten kannalta toiminta jää arvailujen varaan. Tietoturvan kannalta ajateltuna ei voi tietää, tekeekö ohjelma dokumentoimattomia toimenpiteitä.

Jos suljetun koodin ohjelman API-rajapintoja ei ole julkistettu, sen kanssa yhteensopivia ohjelmia on käytännössä mahdotonta tehdä. Julkistettujen API-rajapintojen suljetuttujen koodien ohjelmatkin voivat sisältää virheellisesti toimivia osia. Tässä tapauksessa sen kanssa toimiva ohjelma ei toimi täydellisesti oikein.

Tärkeän suljetun koodin ohjelman valmistaja voi pakottaa käyttäjät käyttämään juuri heidän suljettua ohjelmistoa, jos se ei ole yhteensopiva minkään muun ohjelmiston kanssa. Sinänsä se ei ole muuta kuin taloudellinen haitta. Se vaikuttaa kuitenkin ohjelmiston kehittelyyn sillä tavalla, että vain sen tekijä pystyy kehittämään sitä edelleen. Avoimia ohjelmistoja pystyy kuka tahansa jatkokehittämään.

Suljetun koodin kehittäminen vaatii yhtiöiltä tai yhteisöiltä resursseja silloinkin kun he eivät haluaisi. Tämä ei ole aina mahdollista tai panos-tuotos -ajattelun mukaan järkevää. Esimerkiksi Yahoo on päättänyt julkaista sähköposteja välittävän ohjelman lähdekoodit siksi, että avoimen koodin kehittäjät muokkaisivat siitä yhteensopivamman erilaisten järjestelmien kanssa ja erilaisiin ympäristöihin[8].

Myös avoin koodi on mahdollinen tietoturva-aukoille. Avoimen lähdekoodin Wordpress-bloggausohjelmiston tekijät ilmoittivat 2.3.2007, että kräkkeri on onnistunut ujuttamaan ohjelmiston 2.1.1-versioon tahallisen tietoturva-haavoittuvuuden. Haavan avulla hyökkääjä voi ajaa omaa koodia kohdepalvelimella[9]. Vaara piilee siinä, että haittapiirre on niin nerokkaasti ohjelmoitu, että sitä ei havaitse edes koodia näkemällä. Toinen vaara on siinä, että tehtyä koodimuutosta ei osata epäillä haittakoodiksi, vaan lisätään projektiin sellaisenaan. Jokaisen koodimuutoksen tietoturvatarkistus olisikin liian paljon tarkistusresursseja vaativaa. Haittaohjelmien havaitseminen avoimesta koodista on kuitenkin monin verroin todennäköisempää kuin suljetulle koodille, sillä avointa koodia tarkastelee tuhansia silmäpareja. Ladattavien ohjelmien mukana annettu MD5-tarkistussumma alkuperäisestä jakelusta olisi paljastanut tässä mainitussa tapauksessa asennuspaketin muunnoksen.

Ohjelmakoodien pito suljettuna koodina ohjelman ideoiden salaamisen takia ei ole kyllin hyvä salaamiskeino. Ohjelmien suorituskelpoisista tiedostoista on mahdollista

8 http://www.digitoday.fi/page.php?page_id=14&news_id=200616485&rss=d DigiToday: Yahoo Mailin koodi aukeaa kehittäjille (2.10.2006)

9 http://www.digitoday.fi/page.php?page_id=14&news_id=20075522 DigiToday: Kräkkeri ujutti Wordpressiin tietoturva-aukon. (5.3.2007)

muodostettavissa tietyn tasoista korkean kielen lähdekoodia debuggauksen avulla. Korkean tason kielen konstruoiminen suoritettavasta binäärikoodisesta ohjelmasta ei onnistu täydellisesti. Silti ohjelman toimintakaavioita ja ideoita voi selvitä kyllin hyvin moniin käyttötarkoituksiin. Tällaisia voivat koodin tutkijalle olla: 1) hän pystyy ohjelmoimaan itse vastaavanlaisia ohjelmia, 2) hän voi kopioida osan koodista omaan ohjelmaan hyödyttämään (ymmärtämättä pohjimmiltaan sen toimintaa) ja 3) paljastaa julkisuuteen ohjelmointi-idean.

2.7 Kustannustehokkuus

Avoimen koodin vapaiden ohjelmien käyttö vähentää kustannuksia ensiksi lisenssimaksujen olemattomuutena silloin kun lisenssistä ei peritä maksua. Toisena säästökohteena tulee avoimen koodin Linuxin ylläpidon vaatimien resurssien kustannusten pienempi määrä. Suuri osa tästä säästöstä tulee siitä kun käytössä epävakaaseen tilaan joutuneita ohjelmia ei tarvitse asentaa uudelleen eikä kokonaisen tietokoneen käyttöjärjestelmää tarvitse asentaa uudelleen.

Avoimen koodin ohjelmistot vähentävät kustannuksia myös mainframe-koneissa, joihin ei perinteisesti ole haluttu asentaa Linuxia[10].

2.8 Maapallon uudet kehittyvät alueet

Globalisaatio lisää maapallon tiedon kulkua eri mannerten välillä. Se lisää sovellusten kotoistustarvetta. Ainakin eri merkistöjä tarvitaan kun käytössä ovat latinalaiset, kyrilliset ja arabialaiset merkistöt. Tietokoneet ja niiden myötä sovellukset leviävät maapallolla uusille alueille, joilla ei ole aiemmin käytetty yleisesti tietokoneita. Tällaisia alueita ovat Etelä-Amerikka, Afrikka ja osin Aasia. Avointen ohjelmien kotoistuksen kustannustehokkuuden merkittävyys tulee kasvamaan kun tietokoneet ja ohjelmistoja joudutaan sopeuttamaan yhä laajenevaan määrään eri kieliä ja alueita.

Avoin koodi soveltuu luonnostaan sellaisille alueille, joilla alueen taloudellinen järjestelmä toimii sosialistisin perustein. Ihmiset eivät ole tottuneet omistamaan mitään. Näin ollen kapitalismissa toimiva omistusoikeus ei ole heille käytännön elämässä koettua. Tämä ilmiö huomattiin Neuvostoliitossa 1990-luvulla.

Kilpailua uusista käyttäjistä on niin laite- kuin ohjelmistopuolella. Tietokonealan vahva vaikuttaja Microsoft pyrkii saamaan uusia käyttäjiä ja varsinkin estämään, etteivät he alkaisi käyttää halpoja mikroja ja ilmaisia ohjelmia. Keinoina on ohjelma-

10 Tietoviikko 2.3.2007 s 8. artikkeli Z – se elää!

tuotteiden riisuttujen ja näin halvempien ohjelmaversioiden markkinointi. Esimerkiksi Microsoft Windows XP:stä on saatavana versio XP Starter Edition Brasilia[11].

Tietokoneet tulee olla halpoja ja niiden sähkönsaanti on järjestettävä eri tavalla kuin teollistuneissa maissa on totuttu. Siksi mikrot varustetaan pienellä generaattorilla, jonka kampea veivaamalla generaattori tuottaa tietokoneen käyttämän sähköenergian.

Internetin leviämiselle maapallon uusille alueille ei ole ohjelmistopuolen ongelmia lisenssipolitiikkoineen, sillä protokollat ovat julkisesti avoimia. Internetin hallintaan käytetyt ohjelmat ovat myös pääsääntöisesti avoimia ja vapaita.

11 http://www.digitoday.fi/page.php?page_id=9&news_id=200510428 Halpa-Windows työntyy Brasiliaan open sourcen väliin (9.3.2007)

3 Avoimet dokumentit

Tieto, jota ohjelmilla käsitellään on vähintään yhtä tärkeässä asemassa kuin ohjelmat. On luonnollista, että avoimen lähdekoodin ohjelmien ohella syntyy tietojen rakenteellista avoimuutta. Tiedon rakenteen avoimuus mahdollistaa tiedon siirron järjestelmästä toiseen muuttumattomana. Asiakirja eli dokumentti määritellään ihmisen havaitsemaksi rakenteiseksi määräksi tietoa, jota voidaan siirtää järjestelmästä toiseen yksiköissä. (Fanderl & al., 1992)

Tässä luvussa käsittelemämme standardit ovat tekstuaalisten dokumenttien standardeja. Luvussa 4 käsittelemme muiden tietomuotojen standardeja. Tällaisia tiedon muotoja ovat mm. ääni- ja videokuvat.

Julkisia tunnettuja asiakirjamuotoja ovat Microsoft Corporationin RTF (Rich Text Format), IBM:n RFT:DCA ja DEC:in DCA (Compound Document Architecture). Adobe on kehittänyt PDF (Portable Document Format). PDF sisältää DRM-tekniikan (digitaalinen käyttöoikeuksien hallinta).

Avoimien ja vapaiden tiedostomuotojen puolesta on tehty 14.6.2006 kannanotto ja adressi. Se on nähtävillä ja allekirjoitettavissa osoitteessa

http://sektori.com/uutiset/7179/kannanotto.

3.1 Standardisointi

Avointen dokumenttien standardeja on kehitetty toisistaan riippumatta ainakin kolmessa paikassa. ISO ja CCITT määrittelivät 1990-luvulla Open Document Architecture (ODA) -standardin. OpenDocument-tiedostomuoto ODF on lyhenne sanoista OASIS Open Document Format for Office Applications. Kolmas standardi on ECMA International Standardin kehittämä Office Open XML (jota nimitetään myös OOXML:ksi tai OpenXML:ksi), jota ei ole vielä hyväksytty kansainväliseksi standardiksi.

OASIS ODF-standardi muodostui pitkän kehitys- ja keskusteluvaiheen jälkeen versioksi 1.0. OpenDocument -standardin ominaisuudet sovittiin useiden eri tahojen kanssa. Ensimmäinen virallinen OASIS-tapaaminen oli 16.12.2002. Siinä keskusteltiin standardista ja sen potentiaalisista mahdollisuuksista. OASIS hyväksyi 1.5.2005 OpenDocumentin osana OASIS-standardeja.

ODF:n standardisointiprosessissa oli mukana monia eri kehittäjiä erilaisten toimisto-ohjelmistojen tai niihin liittyvien asiakirjajärjestelmien kehittäjien parista, joita olivat

mm. (aakkosjärjestyksessä): 1) Adobe (Framemaker, Distiller), 2) Arbortext (Arbortext Enterprise Publishing System), 3) Corel (Word Perfect), 4) IBM (Lotus 1-2-3, Workplace), 5) Intel, 6) KDE (Koffice), 7) SpeedLegal (SmartPrecedent enterprise document assembly system); both product and company later changed names to Exari, 8) Novell ja 9) Sun Microsystems / OpenOffice.org (StarOffice/OpenOffice.org)

3.2 ODA

ISO:n ja CCITT:n määrittelemä Open Document Architecture (ODA) dokumenttien muuntoformaatit, esityskonseptit ja muotoiluparametrien merkityksen. Se on malli yhdistetylle tekstiosien, geometristen graafisten elementtien ja rasterigrafiikan muodolle ja esitykselle. Termillä blind document interchange (sokea muunnos) tarkoitetaan sitä, että dokumentti voidaan siirtää kahden järjestelmän välillä siten, että muunnos ja esitys perustuu noudattaa kansainvälisiä standardeja. ODA erottelee toisistaan kolme dokumenttien arkkitehtuuria: 1) formatoidut dokumentit, 2) prosessoitavat dokumentit ja 3) formatoidut prosessoitavat dokumentit. (Fanderl & al., 1992)

ODA määrittelee kolme pääosaa prosessoitaville dokumenteille.: dokumentin sisältö, dokumentin looginen rakenne, esitysmääritykset (säännöt). Looginen rakenne määrittelee esimerkiksi, miten dokumentti on jaettu loogisiin elementteihin kuten luvut, osiot, kappaleet, listat, alaviitteet ja niin edelleen. Looginen rakenne koostuu yleis- ja erityisosasta. Yleisosa voi sisältää esimerkiksi liikekirjeen yleiset määritykset ja erityisosa yleisiä määrityksiä käyttävän osan lisäksi omia määrityksiä.

3.3 ODF

OpenDocument-tiedostomuoto ODF on lyhenne sanoista OASIS Open Document Format for Office Applications[12]. Se on sovelluskehittäjästä riippumaton avoin tiedostomuoto asiakirjoille, joita tallennetaan toimistoympäristössä. Sellaisia ovat teksti- (muistioita, raportteja ja kirjoja), taulukkolaskenta-, kaavio- ja esitysgrafiikka-asiakirjat. Standardin kehitti OASIS industry consortium, ja se perustuu XML-pohjaiseen tiedostomuotoon. Se kehitettiin alun perin OpenOffice.orgia varten. OpenDocument tunnetaan myös ISO-standardointijärjestön nimellä "iso/iec 26300". (ODF, 2005)

Standardi on saatavilla julkisesti. Sitä voi kuka tahansa käyttää ja sisällyttää

12 http://www.oasis-open.org/home/index.php OASIS kotisivu (28.5.2008)

sovelluksiinsa avointen ohjelmistojen tavoin ilman rajoituksia. Standardi luotiin tarjoamaan vapaa vaihtoehto sovelluskohtaisille suosituille asiakirjatyypeille (esim. DOC, XLS ja PPT), joita käytetään Microsoft Officessa. Se on jo nyt sisällytetty useiden ohjelmistokehittäjien ohjelmistoihin ja sen voi kuka tahansa julkaisija liittää omaan ohjelmaansa (sisältäen myös sovelluskohtaiset ohjelmistokehittäjät jotka käyttävät GNU GPL lisenssiä) ilman rajoituksia. Standardi on kehitetty lisäksi kilpailemaan Microsoft Office Open XML -formaatille. Se sisältää näet lisenssi-vaatimuksia ja rajoituksia, jotka estävät useimpia Microsoftin kilpailijoita käyttämästä sitä omissa ohjelmissaan.

Avoimeen tiedostomuotoon tietoja tallentavat yksityiset ihmiset ja organisaatiot eivät ole riippuvaisia yhden ohjelmistokehittäjän talletusmuodosta eikä tuotteesta. He eivät ole pakotettuja lisenssipolitiikkoihin, maksullisiin ohjelmien lisäosiin eikä tuotteen valmistajan tai myyjän poistuminen markkinoilta aiheuta talletusmuodon ja niitä käyttävien ohjelmien kehityksen pysähtymistä. Esimerkiksi henkilö- ja raharesursseja vaativaa ohjelmistojen ja formaattien vaihtotoimenpidettä ei tarvitse tehdä.

Monopoliaseman saavuttaneet kehittäjät ja talletusmuodot eivät voi käyttää asemaansa hyväksi pakottaakseen asiakkaat haluamiinsa muutoksiin, rajoituksiin, lisenssipolitiikkoihin tai maksullisiin päivityksiin tai ohjelmien lisäosiin. Pahimmassa tapauksessa muutos voisi merkitä asiakkaalle vanhojen tietojen muuttumista käyttökelvottomaksi.

W3C on julkaissut oman standardinsa CDF Compound Document Format. The OpenDocument Foundation ilmoitti alkavansa tukea CDF:ää[13]. Marraskuussa 2007 järjestön www-sivut katosivat. Sen katsotaan merkitsevän järjestön toiminnan loppumista.

Avoimia dokumentteja tukevat ohjelmistot

Avoimen dokumenttimuodon käyttöä mahdollistaa useiden ohjelmistojen tukeminen nykyään ODF-standardia: 1) Abiword 2.3, OpenWriter laajennuksen läpi, 2) docvert, web palvelu joka käsittelee tekstitiedostoja (yleensä .doc) ja kääntää ne OpenDocumenteiksi (perustuu OpenOffice.orgiin), 3) eZ publish 3.6, OpenOffice laajennuksen kautta, 4) IBM Workplace, 5) Knomos case management 1.0, 6) Koffice versiosta 1.4 lähtien, joka on julkaistu 21.6.2005, 7) OpenOffice.org 1.1.5 ja 2.0, 8) Scribus 1.2.2, osaa tuoda OpenDocument tekstiä ja grafiikkaa tiedostoista, 9) TextMaker 2005 beta ja 10) Visioo Writer 0.5.2 versiosta lähtien.

13 Odf-kannattajien rivit hajosivat http://www.tietoviikko.fi/doc.te?f_id=1253188 (10.6.2008)

Microsoft Corporation uutisoi 21.5.2008, että Microsoft Office tukee ODF-standardia versiosta 1.1 lähtien Microsoft Office 2007 Service Pack 2 (SP2) -päivityksellä. (Microsoft, 2008) Päivitys julkaistaan vuoden 2009 alkupuolella.

Rakenne

OpenDocument asiakirjoille yleisimmät tiedostopäätteet ovat .odt tekstiasiakirjoille, .ods taulukkolaskennoille, .odp esitysohjelmille, .odg grafiikalle ja .odb tietokantaohjelmistoille. Asiakirja voi sisältää mitä tahansa osia näistä asiakirjamalleista. Asiakirjamallit on tuettu myös OpenDocument muodossa. Mallit sisältävät vain muotoilutietoa (tyyli) asiakirjoille ilman varsinaista tekstisisältöä.

Kuten ylempänä on todettu, OpenDocument voi sisältää tekstiä, taulukkolaskentaa, esitysgrafiikkaa, piirroksia/grafiikkaa, kuvia, kaavoja, matemaattisia lausekkeita, tietokantoja ja niitä yhdisteleviä isäntäasiakirjoja. Se voi myös esittää mallia monille näistä.

OpenDocument-tiedosto on oikeastaan useita erilaisia tiedostoja ja hakemistoja sisältävä ZIP-pakattu arkisto. Tämän ansiosta OpenDocument tiedostot ovat yleensä huomattavasti pienempiä vastaavia Microsoftin .doc tai .ppt muotoisia tiedostoja. Pieni tilantarve on eduksi organisaatioille, jotka säilövät suuria määriä asiakirjoja pitkäksi aikaa, sekä myös organisaatioille joiden täytyy siirtää tiedostonsa hitaan internetyhteyden yli. Kun paketti on purettu osiin, on suurin osa datasta yksinkertaisissa tekstipohjaisissa XML-tiedostoissa, joita voi muokata myös helposti yksinkertaisella tekstimuokkaimella.

OpenDocument tiedostomuoto erottelee selkeästi sisällön, tyylin ja metadatan. Huomattavimmat komponentit tiedostossa ovat selostettuna alla luvussa Rakenne. Tiedostot XML-muodossa ovat määritelty käyttäen RELAX NG kieltä, jotta XML saataisiin hahmoteltua paremmin. RELAX NG itsessään on määritelmänä osa OASIS spesifikaatiota, kuten myös osan kaksi kansainvälistä ISO/IEC 19757: Document Schema Definition Languages (DSDL) määritelmää. Seuraavaksi tarkastelemme avoimen dokumentin arkistopakkauksen rakennetta.

Tiedosto *content.xml* on kaikkein tärkein Open Document -asiakirjan tiedosto. Se sisältää asiakirjan tosiasiallisen sisällön paitsi binääridatan kuten kuvatiedostot. Perusformaattia on inspiroinut HTML. Se on rakenteeltaan se on monimutkaisempaa ja monipuolisempaa. Se on kuitenkin ihmisen luettavissa ja ymmärrettävissä.

Tiedosto *styles.xml* sisältää asiakirjan tyylitiedot. OpenDocument käyttää tyylejä tekstin esitystavan muotoilemiseen ja rikastuttamiseen. Suurin osa asiakirjan ulkoasusta eli tyylistä on tallennettuna tähän tiedostoon (pieni osa on tallennettu

content.xml tiedostoon). Tyyli muotoiluita ovat paragraafin tyyli, sivun tyyli, merkistön tyyli, kehyksen tyyli ja listan tyyli.

OpenDocument määritelmä on vähän epätavallinen, koska siinä ei voi välttää tyylien käyttöä muotoilussa. Ohjelmisto luo tarvittaessa dynaamisesti uuden tyylin.

Tiedosto *meta.xml* sisältää tietoa asiakirjasta tiedostona. Tietoja asiakirjasta on samaan tapaan kuin html-muotoisessa www-sivussa. Esimerkiksi, kirjoittaja, kuka on muokannut, viimeisen muokkauksen päiväys, jne.

Tiedosto *settings.xml* sisältää suurennustason tai kursorin paikan ja vastaavia tietoja. Nämä asetukset eivät ole osana asiakirjan sisältöä tai ulkoasua.

Kansiossa *Pictures/* on kaikki asiakirjan kuvatiedostot. Niihin kaikkiin viitataan content.xml tiedostossa käyttäen <draw:image> -tunnistetta. Se on samanlainen kuin HTML:nn <img> -tunniste:

Asettelutietoa (pituus, ankkuroitu, jne) tarjoaa <draw:frame> tunniste, joka sisältää <draw:image> -tunnisteen.

Useimmat kuvat pidetään niiden alkuperäisessä muodossa (GIF, JPEG, PNG), mutta esimerkiksi bitmap-kuvat käännetään PNG-kuviksi, koska näin saadaan tiedostokoko pienemmäksi.

Tiedostossa *mimetype* on vain yksi rivi, joka kertoo asiakirjan MIME-tyypin. Sillä ilmaistaan tiedoston tyyppi. Koko asiakirjan tiedostopääte (suffiksi) kuten .odt on olemassa vain tietokoneen käyttäjää ja käyttöjärjestelmää varten.

OpenDocument on suunniteltu kokoamaan sateenvarjona jo olemassa olevat avoimet XML-standardit yhteen. Tavoitteena on luoda uusi tiedostomuoto vain siinä tapauksessa, että käytetty olemassa oleva standardi ei pysty tarjoamaan tarvittavaa ominaisuutta tai toimintoa.

OpenDocument käyttää muiden muassa Dublin Corea metadatalle, MathML-skeemaa matemaattisille kaavoille, SVG:tä vektorigrafiikkaan ja SMIL:iä multimediasisällölle.

3.4 OOXML

OOXML on ehdotus taulukkolaskentapohjien, kaavioiden, esitysten ja tekstinkäsittelyasiakijojen ISO-standardiksi. Microsoft on kehittänyt sen alunperin Microsoft Office binääritiedostojen seuraajaksi. Kansainvälinen ECMA on myöhemmin kehittänyt siitä Ecma 376 -standardin vuoden 2006 lopulla. Standardi on

vapaasti saatavana. ISO-standardiehdotus on korjattu versio tästä Ecma 376 -standardista. ISO:n hyväksyntään vaaditaan havaittujen virheiden korjaaminen.

OOXML:n standardointi on tämän tutkielman kirjoitusvaiheessa vielä kesken. 6.6.2008 neljä kansallista jäsenmaata ovat tehneet valituksia ISO/IEC DIS 29500, Information technology Office Open XML formaatista hyväksymisprosessissa ISO/IEC kansainväliseksi standardiksi.

Standardi on saanut aikaan kansalaisliikkeitä kuten esim. Nooxml[14]. Suomessa standardi on jakanut kansalaisia puolesta ja vastaan ryhmiin. Joidenkin arvioiden mukaan standardia kehitetään lähinnä Microsoftin etuja ajamaan. Siitä syystä esimerkiksi yhteensopivuus aiempien teksti- ja taulukkolaskentaformaatteihin on aiheuttanut ongelmia. (Ollila, 2008)

OOXML:ssä tiedostot pakataan Open Packaging Convention (OPC) -menetelmällä ZIP-muotoiseksi pakkaukseksi. Microsoft on kehittänyt OPC:n, jotta XML-muotoinen tieto voitaisi pakata ei XML-muotoisen tiedon kanssa samaan pakettiin. Tällä tavalla tekstien lisäksi voidaan pakata kuvatietoa ja jopa ODF-dokumentteja. Tiedosto sisältää erityisillä merkkauskielillä koodattuja asiakirjan osia. OOXML käyttää moniosaisia sanastoja, joissa on 87 nimiavaruutta ja 87 skeema-moduulia. Merkkauskieliä on tekstinkäsittelyyn, taulukkolaskentaan, esitysgrafiikkaan, vektorigrafiikkaan, kaavioihin. Lisäksi käytetään useita jaettuja merkkauskieliä. (OOXML, 2008)

OOXML:n peruspaketti sisältää XML-tiedoston nimeltään Content_Types.xml juurihakemistossa ja sillä on kolme alihakemistoa: _rels, docProps, ja dokumentin tyypin mukainen hakemisto (esimerkiksi .docx -tekstinkäsittelypaketissa siellä on word -niminen hakemisto). Tämä tyypin mukainen hakemisto sisältää tiedoston document.xml, joka on dokumentin sisältö. Hakemistossa _rels määritellään tiedostojen väliset suhteet paketissa. Juurihakemiston Content_Types.xml määritellään OOXML-dokumentin paketin sisältö. (OOXML, 2008)

14 http://www.noooxml.org/petition Say NO to the Microsoft Office broken standard (10.6.2008)

4 Muita avoimia rakenteita

Wikipedia (englanninkielinen) määrittelee avoimen standardin (open standard) julkisesti saatavana erilaisilla käyttöoikeuksilla varustetuksi standardiksi. Tavallisesti avoimella tarkoitetaan vapaaksi tekijänoikeudellisista maksuista. Tiedon muotoja määritteleviä standardeja sanotaan avoimiksi muodoiksi (open format). Avoimien muotojen pääasiallinen tarkoitus on olla saatavilla pitkän ajan ilman epävarmuutta käyttöoikeuksista tai teknisistä määrityksistä. Avoimia muotoja löytyy osoitteessa

http://en.wikipedia.org/wiki/Open_format.

Tietokoneilla on käsitelty erilaisia tiedonmuotoja kuten ääntä ja videokuvaa. Näiden käsittelemiseksi erilaiset yritykset ovat kehittäneet omia tiedostomuotojaan. Äänentoistossa tunnettuja ohjelmia ovat MediaPlayer, RealPlayer ja QuickTime sekä muutama tuntemattomampi ohjelma. Kaikki nämä käyttävät omia tiedostomuotojaan. Videokuvan käsittelyssä on omat ohjelmat. Tunnettuja protokollia ovat MPEG (Moving Picture Experts Group), AVI (Audio Video Interleave), WMV (Windows Media Video), Applen QuickTime, yms. Digitaalisen TV:n protokollia ovat DVB (Digital Video Broadcasting), ATSC (Advanced Television Systems Committee) ja ISDB (Integrated Services Digital Broadcasting).

Erilaiset tiedostomuodoista johtuen tiedostot ja mediavirrat ovat avautuneet vain sille tarkoitetulla ohjelmalla. Jos käytössä ei ole tarvittavaa ohjelmaa, tiedon muodon voi muuntaa toiseksi muunnosohjelmalla. Haittana näistä on käsittelyn monimutkaistuminen ja mahdolliset tiedon katoamiset. Muunnosprosessissa muodosta toiseen tieto voi muokkautua siten, että alkuperäinen yksityiskohtainen tieto tulee sumeammaksi yms.

Käytön helppouden kannalta tiedostomuodon tulisi olla sellainen, että se avautuu suoraan sovelluksilla. Toiseen luokkaan kuuluvat ne, joita voi avata sovelluksella, mutta pienellä toimenpiteellä kuten avaaminen jossain muodossa. Kolmanteen luokkaan kuuluvat ne, joita voi muuntaa muodosta toiseen jollain muunnosohjelmalla ja sen jälkeen se sitten avautuu suoraan käytetyssä sovellusohjelmassa. Neljänteen luokkaan kuuluvat ne, joita ei voi edes muuntaa eli on etsittävä tiedostomuodon osaava tietokone ja sen käyttöjärjestelmä sovellusohjelmistoineen. Viidenteen luokkaan kuuluvat ne, joita ei voi saada millään tavalla käytön mahdollistavaan muotoon. Tällaisia voivat olla esim. vanhat formaatit, joita ei enää tueta (esim. 1980-luvun PAK-tiedostonpakkaus, jota ei nykyisin ole olemassa eikä siitä ole saatavilla yleisesti tietoa).

Äänentoiston teknologioita

Äänen tallennuksessa ja käsittelyssä avoin vaihtoehto on Ogg Vorbis. Sen kehittämistä on ohjannut Xiph.org -säätiö. Sitä alettiin kehittää vuonna 1998 heti kun MP3-kehittäjä ilmoitti aloittavansa periä lisenssimaksuja. Tarve lisenssittömälle tallennusmuodolle oli suuri. Lopullinen versio Ogg Vorbis -muodosta julkaistiin kesällä 2002.

Vorbis on äänen pakkausmuoto. Pakattu ääni tallennetaan Ogg-muotoiseksi tiedostoksi. Ogg-tiedostoon voi pakata myös muulla pakkaustavalla pakattua ääntä. Siihen voidaan tallentaa esimerkiksi FLAC, Theora ja Speex koodattua ääntä.

Ogg Vorbis -pakkausmuodon äänenlaatu on joidenkin puolueettomien testien mukaan MP3-muotoa parempi. Haittapuolena on pakkauksen purkuun menevä suurempi laitetehovaatimus. Kannettavissa toistimissa tehovaatimukset ovat kriittisiä.

Videokuvan teknologioita

OGM eli Ogg Media Format[15] on avoin ja vapaa muoto videomuotoiselle tallenteelle. Se kehitettiin paikkaamaan AVI-muodon pahimpia puutteita. OGM tukee useita ääniraitoja, useita tekstityskieliä. Pakettimuotoisuuden ja virheenkorjauksen ansiosta se soveltuu hyvin Internetin kautta siirrettäviin videoihin ja videolähetyksiin.

Muut teknologiat

Edellä on esitetty vapaita tietomuotoja ääni- ja kuvamateriaalille. Oma lukunsa ovat ihmisen muut aistit ja niillä aistittavat tiedot. Esimerkiksi tuntoaisti, hajuaisti, makuaisti ja tasapainoaisti aistivat ympäröivää maailmaa kuten näkö- ja kuuloaistikin. Näille ei ole toistaiseksi tiedossa mitään yleisesti käytettyä teknologiaa tai tietomuotoa. Käytetyistä laitteistakaan ei ole mitään tietoa julkisuudessa.

Yleisradio ja avoimet muodot

Viime aikoina on esiintynyt julkista keskustelua[16] paljon käytettyjen palvelujen tiedon muodoista. Esimerkiksi Suomessa Yleisradion (YLE) jakamista äänellisestä ja kuvallisesta mediasta on syntynyt kritiikkiä: miksi niitä voi seurata vain jollain yksittäisellä tietokoneohjelmalla, jonka on valmistanut monopoliasemassa oleva ohjelmistotuottaja. Haittapuolia tästä syntyy kun käyttäjä joutuu hankkimaan kyseisen

15 http://www.faqs.org/rfcs/rfc3533.html RFC 3533 The Ogg Encapsulation Format (13.6.2008)

16 http://blogit.yle.fi/formaattikeskustelu_jatkuu (28.5.2008)

toisto-ohjelman, jos hänellä ei sitä ole jo käytettävissä. Toinen haittapuoli on, että toisto-ohjelmisto voi toimia vain tietyllä tietokonearkkitehtuurilla. Jotkut näkevät, että on YLE:n henkilökunnan osaamattomuutta tai haluttomuutta oppia uutta se, että avoimiin tiedostomuotoihin ei ole jo siirrytty.

Keskustelussa on miellyttävänä piirteenä YLE:n mukanaolo. He ovat vastanneet annettuihin kommentteihin. Keskustelu voidaan jakaa formaatin edistyksellisyyteen (tiedon katoaminen, tietoturva, lisensiointi, suunnattu media, yms.), ohjelmistojen toimivuuteen (Firefox kaatuu, yms.) ja Microsoftin syrjimiseen tähtäävään keskusteluun. Keskustelussa on tullut runsaasti asiaa avointen ohjelmien näkökulmasta.

Keskustelu sinänsä on hyvä esimerkki avoimien tiedostoformaattien ja samalla avoimien ohjelmien käytön leviämisestä. Sehän osoittaa, että niitä käytetään kun ihmiset puhuvat niihin sopivista tiedostomuodoista.

Yleisradion Elävän Arkisto julkaisee vain Windows Media -muotoista mediaa (tilanne vuonna 2007). Elävää arkistoa pidetään koko kansan arkistona. Sen tulisi olla kaikkien käytettävissä. Tilannetta parantaisi huomattavasti, jos Elävä Arkisto julkaisisi tarjolla olevan median esimerkiksi myös Real Player -muotoisena tai nimenomaan avoimella standardilla Ogg Vorbis tai Ogg Theora –muotoisena.

Keskustelu avoimista mediamuodoista on myös hyvä esimerkki asiakaspalvelun avoimuudesta. Valtion yhtiö YLE saa palautetta ja opastusta nettikeskustelujen kautta ja käyttää annettuja kommentteja hyväksi kehitystyössään. Yleisradio saa tässä julkisessa keskustelussa jopa ammattitaitoa ilmaiseksi, jota jouduttaisiin hankkimaan muutoin (kalliilla) kursseilla työpaikan maksamana tai ostopalveluina.

Avoimissa mediateknologioissa ei ole otettu huomioon tiedon suojausta. Sellainen ominaisuus on kuitenkin tarpeen, jotta teknologia voisi yleistyä kaupallisten mediatuotteiden tallennus- ja käsittelymuodoksi. Keskusteluissa on pohdittu, voiko avoimiin tiedostoformaatteihin saada DRM-suojausta[17] tai sen tasoista vastaavaa suojausta. Kun on media voidaan suojata kunnollisesti, se voidaan julkaista sellaistakin materiaalia, jota ei ole aiemmin saanut julkaista Internetissä kaikille vapaasti. Esimerkkinä tällaisesta suojauksesta on Media Playerin DRM. Keskustelussa on otettu myös se näkökulma, että suojaus estää vain laillisesti hankittujen medioiden käyttöä. DRM ei kuitenkaan pysty estämään luvatonta kuitenkaan hakkerointia.

17 http://en.wikipedia.org/wiki/Digital_Rights_Management Digital Rights Management (29.5.2008)

5 Avoimien ohjelmien ohjelmistotuotanto

Ohjelmistotuotanto on ohjelmistojen suunnittelua, kehitystä ja ylläpitoa järjestelmällisesti ja kurinalaisesti. Kehitysprosessiin kuuluu seuraavia osia: vaatimusmäärittely, järjestelmäsuunnittelu, ohjelmistosuunnittelu (toiminnallinen ja tekninen määrittely), toteutus, testaus, julkistus ja ylläpito. Tämä kuvaa myös ohjelmiston elinkaaren. Ohjelmistotuotanto sivuaa tietokonetuotantoa, tietojenkäsittelytiedettä, hallintaa, matematiikkaa, projektinhallintaa, laadun hallintaa, ohjelmien ergonomiaa ja järjestelmätuotantoa. (Haikala & Märijärvi, 1997) (Wikipedia, 2008) Ohjelmistotuotanto sivuaa myös ekonomiaa tuotannon taloudellisessa tehokkuudessa.

Ohjelmistotuotannon työkaluja ovat olio-ohjelmointiin kehitetty UML-mallinnus (Unified Modelling Language). Se on yhtenäistetty mallinnuskieli, jolla suunnitelmia visualisoidaan. Se koostuu yksinkertaisista graafisista kuvioista ja niiden välisistä yhteyksistä. CASE (Computer Aided Software Engineering) on vanha ohjelmisto-tuotannon prosessien visualisointi ja organisointityökalu. Eri ohjelmointikielille ja -ympäristöihin on omat CASE-työkalunsa.

Avoimien ohjelmien tuotanto noudattaa yllämainittua ohjelman tai ohjelmiston elinkaarta. Se eroaa perinteisestä suljetun koodin ohjelmien kehityksestä siinä, että kehittely on avoimuuden lisäksi hajautettua. Hajautus tarkoittaa tässä yhteydessä eri yrityksiä tai yksityisiä henkilöitä eikä pelkästään yhden ohjelmistoalan yrityksen eri osastoja tai sisaryrityksiä. Kullakin avoimien ohjelmien kehittäjällä voi olla omia intressejä ohjelman ominaisuuksista ja myös teknisestä toteutuksesta. Avointen ohjelmien kehityksen hajautus tarkoittaa lähinnä toteutusvaiheen hajautusta.

Avoimien ohjelmistojen tuotannon ympäristöön kuuluu keskeisenä versionhallinta-järjestelmä, joka kokoaa suuren määrän kehittäjien tekemiä muutoksia järjestelmällisesti arkistoksi. Versionhallinnasta eri kehittäjät saavat myös viimeisimmät koodiversiot ohjelmasta. Versionhallintajärjestelmässä muutoksista pidetään kirjaa tehdyistä muutoksista. Tarvittaessa (kun havaitaan virheitä) voidaan palata aiempiin kehitysversioihin helposti. Versionhallintajärjestelmä pitää myös tilastoa.

Ohjelmistojen kehityksen päämäärä ovat uudet innovaatiot, vakaat vikasietoiset ohjelmistot ja tietoturvallisuus. Esimerkkinä uusista innovaatioista vuonna 2006 on hypermediamainen työpöydän ikkunointijärjestelmän tulo Linux-käyttö-järjestelmälle[18]. Siinä ikkunasta avattu toinen ikkuna muodostaa hyperlinkin

18 http://www.coss.fi/web/coss/developers/summercode/2006 COSS kesäkoodiprojekti 2006

edelliseen. Näin muodostuu hyperverkko ikkunayhteyksiä (joka on rakenteeltaan puu, sillä uutta ikkunaa ei voi avata jo olemassa olevana). Linux-käyttöjärjestelmän työpöytäohjelmissa ikkunointi on ollut perinteisesti lineaarinen lista.

Avoimen koodin ohjelma joutuu monivaiheiseen prosessiin ennen loppukäyttäjälle saapumista. Koodi on viety SVN-versionhallintajärjestelmään (SVN, 2000), jossa sitä voidaan kehittää keskitetysti hajallaan olevien kehittäjien välillä. Koodista käännetään suoritettavien ohjelmien paketteja (binääripaketteja) ja valmistetaan versioita erilaisiin käyttöjärjestelmiin (siirros) ja käyttöympäristöihin. Jokainen eri käyttöympäristöön muokattu paketti voi sisältää omia laajennuksiaan, joita ei ole alkuperäisessä koodissa. Kotoistuksessa ohjelmat käännetään eri kielelle ja eri käyttöjärjestelmille versioiduissa siirroksissa kielikäännökset ovat erilaiset, sillä jossain versiossa voi olla enemmän tekstiä kuin toisessa. Kotoistuksen ongelmia eri Linux-distribuutioiden välillä on havaittu KDE:n kotoistuksen ja Rosettan välillä[19].

Ohjelma on käännettävä lähdekoodista suoritettavaksi ohjelmaksi (compile) jokaisen muutoksen jälkeen, tehtävä siirros uudelleen ja käännettävä eri kielille (translate), ennen kuin se on valmis loppukäyttäjälle jaettavaksi. Tällainen vaatisi valtavan määrän käsin tehtävää työtä, jos sitä ei automatisoida. Kääntäminen ohjelman lähdekoodista eri siirroksille on triviaalia komentotiedostojen avulla.

Ennen vuosituhannen vaihdetta 2000 ohjelmistojen jakelu oli hajanaista ja altis virheille (Hoek & al., 1997). Vuonna 2006 lähes jokaisessa käyttöjärjestelmässä on paketinhallintajärjestelmä, joka pitää kirjaa järjestelmässä olevista ohjelmapaketeista ja niiden versioista ja osaa tarvittaessa ehdottaa päivityksiä. Päivitykset ovat myös lähes automaattisia. Nyt kehityksen kohteena on eri siirrosten välinen automaattinen hallintajärjestelmä (tämä on kirjoittajan oma arvio).

Avoimen ohjelmakoodin kehittäjä antaa työnsä tulokset julkisesti saataville. Tämä tarkoittaa mm. sitä, että hän luopuu mahdollisesta taloudellisesta hyödystä ja antaa innovaationsa toisten käyttöön. Avoimien ohjelmien määrittelyssä tekijänoikeus ohjelmakoodiin säilyy kuitenkin tekijällä. Onko luopuminen taloudellisesta hyödystä kehittäjän kannalta mitään järkeä? Avointen koodien kehittäjät saavat elantonsa muualta, joten se ei ole heille elintärkeä asia. Avointen ohjelmien kehittelyyn pätee ajatus: kun minä annan oman panokseni yhteiseen hyvään, joku toinenkin ja kolmas antaa ja me kaikki hyödymme tuloksista. Usein syntyy myös kumulatiivista lisäarvoa.

19 https://wiki.kubuntu.org/RosettaAndUpstreamCollaboration Rosetta and Upstream Collaboration (13.10.2006)

5.1 Ohjelmistokehitys

Lähdekoodin avoimuus ei muuta sinänsä ohjelmakoodia. Onko avoimen lähdekoodin ohjelmien kehittäminen samanlaista perinteisen suljetun koodin kanssa? Avoimen lähdekoodin ohjelmien suunnittelu on saanut aikaan kohun ohjelmistokehityksessä. (Spinellis & Szyperski, 2004)

Avoimen lähdekoodin ohjelmistotuotannossa kehittäjä suunnittelee ohjelmistoja valmiiden olemassaolevien ohjelmistoelementtien pohjalta. Pyörää ei tarvitse keksiä uudelleen. Valmiit elementit (komponentit) laadukkaita. Ne toimivat tehtävässään parhaiten, ovat nopeita ja turvallisia sekä valmiiksi testattuja. Valmiit ohjelmistoelementit ovat tyypillisesti täydellisiksi kehitettyjä. Toinen etu avoimen lähdekoodin suunnittelussa on se, että lähdekoodiin voidaan tehdä muutoksia. Muutokset voivat olla muutaman rivin suuruisista muutoksista jopa kokonaiseen ohjelmakomponenttiin. Kolmanneksi koodi voidaan siirtää (porttaus) helposti haluttuun kohdejärjestelmään toimivaksi.

Ohjelmistojen suunnittelijat voivat tehdä muutoksia myös alkuperäiseen ohjelmakoodiin käyttäessään niitä valmiita ohjelmakomponentteja. Tällaisia koodin parannuksia voivat olla esimerkiksi turvallisuuteen liittyvät parannukset ja uusien ominaisuuksien lisäykset.

Koodin uudelleen käytöstä ja alkuperäisen koodin muutoksilla voi olla myös haittoja. Uudelleenkäytetty ohjelmaelementti voi sisältää osia, joilla ei ole minkäänlaista toiminnallista merkitystä uudessa tehtävässä. Silloin ylimääräinen koodi on turhaa ja jopa turvallisuusriski. Alkuperäistä koodia muuttava voi saada aikaan vastaavanlaisia ominaisuuksia siihen koodiin.

Avoimen lähdekoodin kehitystyöprosessissa on nähtävissä seitsemän erityispiirrettä. 1) Eri osien kehitys tapahtuu enemmän rinnakkain kuin lineaarisesti edeten yhtenä kehityskohtana. Tämä lisää kehityksen nopeutta valtavasti. Haittapuolena voi olla toisistaan tietämättömät kehittäjät, jotka kehittävät samaa asiaa. 2) Kehittäjiä on paljon ja he ovat maantieteellisesti hajallaan. 3) Vertaisarviointi toteutuu avoimen ohjelmistojen kehityksessä hyvin. 4) Käyttäjien palautteilla ja kehittäjillä on läheinen yhteistyö. 5) Mukana on älykkäitä ja motivoituneita osaavia kehittäjiä. Oman kehitystyönsä ohella he ovat innostava esikuva muille kehittäjille. 6) Ohjelmien käyttäjät voivat olla jopa mukana kehitystyössä. 7) Avoimen lähdekoodin ohjelmilla on erittäin nopea julkistus. (Feller & Fitzgerald, 2001)

Software Freedom Law Centerin perustaja, puheenjohtaja ja toiminnanjohtaja Eben Moglen sanoo ComputerWorld -lehden helmikuun numerossa 2008 avoimen lähdekoodin ohjelmia uusiutuvaksi luonnonvaraksi. Kuka tahansa voi luoda uutta koodia

käyttämällä hyväksi jo olemassaolevaa koodia.

Avoimen lähdekoodin ohjelmien kehityksessä tutkielman luvun 5.2 ohjelmien komponenttirakenne ja siitä johtuvat ongelmat tulevat esille. Ohjelmistokehityksessä on erittäin helppo luoda syviä riippuvuussuhteita komponenttien välille. Tästä seuraa ongelmia kun komponentteja käytetään uudelleen uusiin ympäristöihin. Riippuvuussuhteista syntyy myös ongelmia kun ohjelmistosta kehitetään uusia versioita. Sitä lisää avointen ohjelmistojen tiheä uusien versioiden ilmestyminen. Tämä voi johtaa anarkistiseen riippuvuussuhdeverkostoon (puumaisista riippuvuus-suhteista) avoimeen lähdekoodiin, käännettyihin kirjastomoduuleihin, paketoituihin komponentteihin, jaettuihin kirjastoihin, header-tiedostoihin, malleihin, tieto-kantoihin, liitännäisiin ja kokonaisiin ohjelmiin. (Spinellis & Szyperski, 2004) Ratkaisu tai ainakin lievitys näihin ongelmiin saadaan kun ohjelmakehityksessä pyritään minimoimaan riippuvuudet komponenttien välillä.

Kehitystyön mallinnus

Ohjelmistokehityksen mallinnus tarkoittaa avoimen lähdekoodin ohjelmien kehityksen ilmiöiden rakenteiden jäsentämistä. On olemassa useita käyttökelpoisia malleja, mutta mikään niistä ei ole hyvä. Feller ja Fiztgerald ottavat esille kaksi mallia.

Zachmanin mallissa on kuusi kuvailevaa kategoriaa: 1) materiaalinen, 2) toiminnallinen, 3) alueellinen, 4) henkilöllinen, 5) ajallinen ja 6) tarkoituksellinen. Materiaalinen kuvausmalli kuvaa, mistä osista rakenne koostuu. Toiminnallinen kuvaus ilmaisee, millä tavalla asiat toimivat. Alueellinen kuvaus kuvaa, missä asiat tapahtuvat. Henkilöllinen kuvaus kuvaa ihmisten suhteita järjestelmään, kuten käyttöliittymiä. Ajallinen kuvaus kertoo, milloin mikäkin asia tulee tapahtua. Tässä käytetään avuksi tilakaavioita. Tarkoituksellinen kuvaus kertoo, miksi mikin asia on olemassa.

Toinen Fellerin ja Fitzgeraldin esille ottama malli on CATWOE. Malli sisältää seuraavat näkökulmat: 1) Sisäiset tai ulkoiset kehitystyön vaikuttamat asiakkaat, 2) järjestelmän sisäiset toimijat, 3) muunnos sisääntulevista tiedoista uloslähteviin tietoihin, 4) maailmankuva, 4) järjestelmän toimivuudesta huolehtivat omistajat ja 6) järjestelmää ympäröivä ympäristö.

Avointen lähdekoodien ohjelmien kehitystyön elinkaari eroaa perinteisestä ohjelmien kehittämisen elinkaaresta (SDLC). Avoimen lähdekoodin kehityskaari noudattaa sykliä: koodin haku, koodin arviointi, esitestaus, kehitysjulkaisu, rinnakkain tapahtuva vianetsintä, tuotejulkaisu. (Feller & Fitzgerald, 2001)

5.2 Työkaluja

Avoimen lähdekoodin ohjelmien kehittäminen on johtanut käyttämään ohjelmisto-kehityksessäkin monipuolisia avoimen lähdekoodin hienoja työkaluja ja suunnittelu-järjestelmiä aina integroituihin ympäristöihin. Työkaluista mainittakoon GNU/Linux, MySQL, JBoss ja GNU Compiler. Integroituja ympäristöjä ovat mm. Eclipse ja Kdevelop. Tällöin muutaman suunnittelijan pienet ryhmätkin voivat käyttää projekteissaan näitä laajoja hyviä työkaluja ja ympäristöjä. (Spinellis & Szyperski, 2004)

Tigris on avoimen ohjelmien kehitystyökalujen yhteisö ja sivusto, jonne on keskitetysti tuotu erilaisia ohjelmankehittämisessä käytettyjä työkaluohjelmia, ympäristöjä ja muita apuohjelmia. Työkaluja löytyy monipuolisesti mm. seuraaviin kategorioihin: 1) automatisoitu ohjelman lähdekoodin analysointi, metriikka ja näihin liittyvät työkalut, 2) koodaus, testaus ja virheenkorjaustyökalut (debuggaus), 3) julkistuksien ja päivitysten ohjelmatyökalut, 4) ohjelmistosuunnittelun työkalut, 5) kehitysversioiden virheiden seurantajärjestelmät ja niihin liittyvät työkalut, 6) uudelleenkäytettävien ohjelmakomponenttien kirjasto, 7) projektinhallintaohjelmat ja työkalut, 8) ammattimaisen ohjelmistokehityksen resurssit, 9) ohjelman vaativuuksien ja riippuvuuksien hallintatyökalut, 10) asetusten hallinnan työkalut (CM), 11) teknisen kommunikoinnin ja yhteydenpidon välineet ja 12) testauksen työkalut. (Tigris, 1997)

5.3 Versionhallinta

Ohjelmistojen komponentit eivät ole itsenäisiä toisistaan riippumattomia kokonaisuuksia, vaan niiden välillä on paljon riippuvuussuhteita. Van der Hoek & al. (Hoek & al., 1997) nostavat esille yhtenä tuotejulkistuksen pääajatuksena ohjelmistotuotteiden komponenttien väliset yhteydet ja niiden kasvavan monimutkaisuuden. Matemaattisesti joukon riippuvuussuhteiden maksimimäärä on 2^n, missä n on komponenttien kappalemäärä. Ohjelmakomponenttien riippuvuus-suhteiden hallittu ylläpito on lähes mahdotonta ihmiselle. Kun otamme huomioon komponenttien eri versiot, määrä moninkertaistuu.

Ohjelmien yhteensopivuusvaatimukset yhä useampien muiden ohjelmien kanssa lisäävät niiden kompleksisuutta. Ohjelmien eri versioiden välillä on riippuvuuksia. Sovellus voi vaatia komponentiltaan vähintään tiettyä versionumeroa, josta lähtien löytyy tarvittu toiminto. Versionumeroehto voi vaikuttaa toisinpäinkin: tarvitaan tiettyä versionumeroa aikaisempi komponentti, jos uudempi ei sisällä tarvittavaa toimintoa. Tällainen tilanne syntyy, kun toiminto siirretään toiseen ohjelma-komponenttiin (joka esimerkiksi saattaa muodostaa loogisesti paremman kokonai-suuden) tai toiminto lohkaistaan omaksi komponentikseen. Mahdollisuuksia on

lukematon määrä komponenttien riippuvuussuhteiden hallinnan kannalta katsottuna.

Versionhallinta on tärkeä osa avointen ohjelmien kehitystyötä. Se on merkittävämmässä asemassa kehitysympäristön hajanaisuuden vuoksi verrattuna suljettujen ohjelmakoodien kehitysympäristöihin. Jokaisen avoimen ohjelmakoodin kehittäjän tulee olla selvillä kehitystilanteesta. Avoimessa kehitysympäristössä tiedon kulku on epämääräisempää ja siksi sen tulee olla näkyvää sidosryhmien välisien kommunikoinnin onnistumiseksi. Suljetuissa ympäristöissä tiedonkulku voi estyä kaikkien osapuolien välillä. Puuttuvaa tietoa ei edes osata välttämättä odottaa, mikä on kohtalokkainta virheiden kannalta katsottuna.

SVN (SVN, 2000) on uudehko versionhallintaohjelmisto, joka on syrjäyttämässä perinteisen CVS-versionhallintaohjelmiston. SVN:n oppaita löytyy Internetissä osoitteesta

http://svnbook.red-bean.com/.

Riippuvuudet tulevat esille avoimen lähdekoodin ohjelmistojen suunnittelussa, kuten luvussa 5.1 on esitetty. Riippuvuudet tulevat esille myös valmiissa ohjelmisto-tuotteissa kuten esimerkiksi Linuxin eri distribuutioiden yhteensopivuuksissa. Uusi ohjelma on toimittava jokaisessa distribuutiossa. Distribuutioiden määrä on kymmeniä.

5.4 Kotoistus

Tietokoneohjelmien kotoistus eli lokalisointi tarkoittaa niiden asettamista eri kielten, kansallisuuksien ja maanosien välillä. Tietokoneohjelmien tärkein kotoistettava ominaisuus on kieli. Muita kotoistettavia asioita ovat tekstin vierityssuunta, päivämäärän esitysmuoto, valuutta, desimaalierotin ja värit. Ohjelmien käyttöoikeus-sopimukset olisi hyvä lokalisoida paikalliseen lainsäädäntöön sopiviksi. Ohjelman yleiset käyttösopimusehdot voivat olla ristiriidassa paikallisen lainsäädännön kanssa. Silloin pitää tarkastella, voiko ohjelmaa edes käyttää siinä paikassa. Tämä kysymys tulee esille enemmän ohjelmilla käsiteltävän tiedon kanssa. Onko tieto kielletty paikallisessa kotoistuksessa, ja onko onko se kriminaalista. Edelleen onko tiedon käyttöä sanktioitu.

Päivämäärän lisäksi kellonajan aikavyöhyke (timezone) tulisi kotoistaa. Joillakin alueilla käytössä on esimerkiksi erillinen kesäaika, joka pyrkii korjaamaan kellonaikaa auringon nousun ja laskun mukaan. Näin tietokonesovelluksen käyttäjän ei tarvitse asettaa kellonaikaa talvi/kesäaikaan kaksi kertaa vuodessa. Säätö voidaan toteuttaa tietenkin aikapalvelimen aikaan täsmäämällä (NTP-protokolla). Edelleen

silloinkin sovelluksessa voi olla tarvetta talvi- ja kesäajan erikoiskäsittelyyn.

Kotoistusta helpottaa kun ohjelmien suunnitteluvaiheessa ohjelmista tehdään sellaisia, että ne voidaan lokalisoida ohjelmasta erillään. Tätä sanotaan internationalisoinniksi. Internationalisoidussa ohjelmassa ohjelman tekstejä ei ole kovakoodattu ohjelma-koodiin, vaan ne haetaan erillisestä moduulista. Moduulia vaihtamalla ohjelman kieli saadaan vaihdettua kätevästi toiseksi.

Sovelluksessa virheilmoitusten kotoistaminen on tärkeä asia. Usein tietokoneita vähän käyttänyt ihminen ei kykene selvittämään eikä korjaamaan sovelluksen käytössä tapahtunutta virhettä. Tämä johtuu hyvin usein siitä, että käyttäjä ei ymmärrä vieraskielistä virheilmoitusta ja ohittaa sen hänelle merkityksettömänä. Tämä johtaa sitten tukihenkilöiden kuormittamiseen kun vian diagnosointiin menee huomattavasti pidempi aika täsmällisen virheilmoituksen puuttuessa. Olen havainnut usein juuri tämän virheilmoituksen lukutaidottomuuden olevan syy, miksi tavallinen tietokoneenkäyttäjä ottaa yhteyttä mikrotukeen tai sen puuttuessa tuttuihin.

Varsinaisten sovellusten lisäksi sovellusten käyttöoppaat tulee kotoistaa. Myös mahdollinen online-help tulee kotoistaa. Tämä on tavallisesti www-sivusto. Sovellusten rekisteröintikaavakkeet tulee kotoistaa olivat ne sitten paperisessa tai sähköisessä muodossa. Sovelluksen sertifikaatit on myös kotoistettava. Avointen ohjelmien kehittely-ympäristöissä nämä kotoistetaan samaan tapaan kuin varsinaiset sovellukset.

Avointen ohjelmien kehitys ja kotoistusympäristössä on tavallisesti useita toisistaan riippumattomia palvelinkoneita. Kahden kielitiedostopalvelimen tilanteesta voi johtua sellainen ristiriitainen tilanne, että tehty muutos ei leviä toiseen paikkaan ollenkaan. Esimerkiksi suosittua työpöytäohjelmaa KDE jaetaan useassa eri paikassa erilaisin siirroksin. Ubuntun KDE kielipaketteja muokataan Rosetassa kun alkuperäinen paikka kielipaketeille on i18n.kde.org. Kun KDE:n kielipakettiin tehdään muutos Rosetassa, muutos ei tule voimaan alkuperäisessä paikassa. Toisinpäin muutosten päivitys toimii viiveellä. Tähän ristiriitaan ei ole kehitetty mitään ohjeistoa. Ongelma on tiedostettu (tilanne syksy 2006) ja ratkaisua odotellaan.

Sovelluksen kotoistus ei ole ainutkertainen tapahtuma sovelluksen eliniän aikana. Sovelluksesta kehitetään ja julkaistaan uusia versioita. Ne on myös kotoistettava.

Sovellusten ja www-sivustojen lisäksi Internetin domain-nimet ovat tekstimuotoista tietoa. Merkistönä on ollut ASCII. Vuonna 2004 syksyllä ICANN aloitti nimien kansainvälistämiskokeilun[20]. Pyrkimys on saada domain-nimetkin kotoistettaviksi.

20 http://www.digitoday.fi/pdf/newsPdf.php?news_id=200725099 (11.10.2007)

Hyötyjinä ovat eniten erilaisia merkistöjä käyttävät kielet kuten arabia, kiina, yms. Näiden alueiden Internetin käyttäjille

Kieli

Suuri osa tietokoneen käyttäjistä osaa tietokonemaailman yleisimmin käytettyä kieltä englantia sen verran, että pystyy käyttämään tarvitsemiaan ohjelmia. Kuitenkin monet, varsinkin vanhemmat ihmiset eivät osaa muuta kuin äidinkieltään. Ohjelmia käyttäessä he oppivat löytämään valikoista oikeat kohdat ilman termien ymmärtämistä. Samalla he oppivat ohjelman käyttämää englantia.

Ohjelmien kotoistuksen tarve tulee näkyviin kun käyttäjän äidinkieli poikkeaa tietokoneiden englannista niin paljon, että päättelemisellä käytön oppiminen ei onnistu. Tällaisia kieliä puhutaan äidinkielenä esimerkiksi Arabiassa, Kiinassa ja Venäjällä. Lokalisoitujen ohjelmien käytön oppiminen on nopeampaa kun ohjelmien kieli on käyttäjän äidinkieli.

Kieli on tärkeä osa ihmisen identiteettiä[21]. Englannin kielen käytön yleistyessä kotoistus vaikuttaa eri kielten säilymiseen kun käyttäjät voivat käyttää tietokoneita omalla kielellään. Kotoistus on laajemminkin kansallisia kulttuureja säilyttävää.

Suomalaisen kotoistamisen koordinoija toimii nykyisin CSC (the Finnish center for science, tieteellisen laskennan palvelu). Aiemmin vetäjänä on toiminut tietotekniikan kotoistushanke Kotuksen[22] koordinoimana. (Kotoistus, 2004)

Kotoistuksessa tulee esille ohjelmistojen eri distribuutioiden välillä ristiriitoja. Saman ohjelman siirroksia eri ympäristöihin käännetään joskus erillisissä paikoissa. Esimerkiksi Debian-pohjaisen Kubuntu-distribuution KDE työpöytäohjelmistoa käännetään sekä Rosetassa, että KDE:n omalla i18n.kde.org -palvelinkoneella. Käännösten synkronointi hoituu siten, että käännöksiä viedään myös manuaalisesti KDE:n palvelimelta Rosettaan (tilanne vuonna 2006). Uudet käännökset ilmestyvät Rosettaan pienellä viiveellä. Joissakin tapauksissa näissä kahdessa kielipakettien luontipaikassa on erilaiset käännökset yksittäisestä ohjelmasta. Tämä voi aiheuttaa käyttäjässä epätietoisuutta, jos hän käyttää samaa ohjelmaa usealla distribuutiolla tai neuvoo tai kysyy neuvoja toisen distribuution käyttäjältä.

Tietokoneohjelmien kielessä käytetään kunkin kielen ns. virallista yleiskieltä. Sitä ymmärtävät periaatteessa kaikki sitä kieltä äidinkielenään käyttävät tietokoneen käyttäjät. Ohjelmia voidaan kääntää myös eri murteille ja erilaisiin sosiaalisiin

21 http://www.finlit.fi/oppimateriaali/kielijaidentiteetti/ Kieli ja identiteetti (8.1.2007)

22 http://www.kotus.fi/ Kotuksen kotisivu (24.9.2006)

ympäristöihin. Suomen kielen murteista esimerkkinä on 1990-luvun alkupuolella ilmestynyt ”savonkielinen DOS”.

Kulttuurillisesti erilaisiin ympäristöihin lokalisoituja tekstejä www-palveluna on esimerkiksi Korsoraattori, joka muuntaa suomen kielen ”nuorison ymmärtämään muotoon”[23]. Tekstin sekaan lisätään kirosanoja ja käytetään muutenkin kieltä. Tällaisen muuntimen toimintaperiaate on annetun tekstin jäsennys ja sen jälkeen muuntaminen tulosteeksi. Idea on levinnyt suomen kielen murteisiin. Internetistä löytyy palveluja, jotka muuntavat suomen kirjakieltä savoksi, Turun murteeksi, Tampereen murteeksi, Oulun murteeksi ja Kannaksen Karjalassa käytetyksi murteeksi. Näillä palveluilla on merkitystä lähinnä ihmisten hauskuttamisessa.

Kielen kääntämisen automatisointi tekisi kotoistuksen helpoksi. Ei ole kuitenkaan olemassa täysin varmoja käännösohjelmia, jotka osaisivat kääntää termit ja lauseet semanttisine merkityksineen oikein kaikissa mahdollisissa tilanteissa. Tämä johtaisi joissakin tilanteissa tekstejä, joita on vaikea ymmärtää ja tekstejä, jotka ymmärretään väärin. Tekstin kääntämistä on mahdollista automatisoida puolittain kaksivaiheisesti. Ensin teksti käännetään koneellisesti ja sen jälkeen ihminen tarkistaa tuloksena syntyneen käännöksen. Tässä on kuitenkin vaara, että tarkistajana ei huomaa virheellistä käännöstä kun hän selaa tekstiä. Toinen epäkohta on resurssien kulutus: tarkistaminen voi viedä resursseja lähes yhtä paljon tai jopa enemmän kuin kääntäminen alkuperäisestä tekstistä ilman koneellista esikääntämistä.

Tekstien kielen kotoistuksen lisäksi kotoistettavaa on oikeinkirjoituksessa, tavutuksessa ja thesaurus-toiminnossa. Näitä tarvitaan oikeastaan vain tekstinkäsittelyohjelmissa ja niissäkin vain ohjelman käsittelemien tietojen käsittelyssä. Oikeinkirjoitussäännöt vaihtelevat eri kielissä. Kielen oikeinkirjoituksen kotoistusta tarvitaan sovelluksissa silloin kun sovelluksen tekstit muuttuvat dynaamisesti ja ajonaikaisesti. Kotoistettu staattinen teksti ei tarvitse oikeinkirjoituksen kotoistusta. Tavutussäännöt vaihtelevat kielestä toiseen. Jos sovellus tavuttaa tekstinsä, silloin tavutuksen kotoistus on tarpeen. Thesaurusta käytetään vain lähinnä tekstinkäsittelyohjelmissa.

Värit ja kuvat

Väreillä on informatiivista vaikutusta. Punaista pidetään lämpimänä värinä. Se virittää ihmiseen sen suuntaisia tunne-elämyksiä. Sininen koetaan viileänä. Nämä tuntemukset perustuvat fysiologisiin aisteissa tapahtuviin reaktioihin.

Näiden tunneperäisten kokemusten lisäksi väreillä on merkityksiä. Nämä merkitykset

23 http://korsoraattori.evvk.com/ Korsoraattori (29.5.2008)

eroavat kansallisuuksien mukaan.

Valkoinen tunnetaan esimerkiksi länsimaisissa kulttuureissa puhtauden ja valon värinä. Musta tunnetaan surun, pimeyden ja pahan värinä. Kuitenkin Aasian maissa merkitykset ovat päinvastaisia. Näin ollen tietokoneohjelmiin tarvitaan värien kotoistusta, jotta ihmisille muodostuu käyttöympäristössään oikea kuva tai tunne ohjelmasta tai sen yksittäisestä toiminnosta.

Myös kuvakkeet ja kuvat voivat vaatia kotoistusta. Eri kulttuureissa kuvilla on eroavuuksia mielikuvien synnyttäjänä. Esimerkiksi ohjelman käytössä poistu, avaa uusi, kopioi jne. toiminnoilla voi olla kussakin kulttuurissa omat kuvakkeet, joista käyttäjä mieltää heti ensi silmäyksellä toiminnan laadun.

Muut kotoistettavat asiat

Muita kotoistettavia asioita ovat päivämäärän esitysmuoto, valuutta ja desimaalierotin.

Päivämäärän esitysmuoto on tärkeä lokalisoitava, koska monissa eri lokaatioissa kuukauden ja päivämäärän paikat ovat käänteiset. Tämä voi aiheuttaa sekaannusta sellaisissa tilanteissa, että numeroarvoista ei voi päätellä, onko se päivä vai kuukausi. Esim. kuukausi 13 on mahdoton. Vaikka oikean päivämäärän pystyisikin konstruoimaan virheellisestä lokalisoidusta päivästä, siihen kuluu aikaa ja mielenkiinto kohdistuu siihen tehtävään pääasian jäädessä taka-alalle peräti keskeytyen.

Valuuttamerkin kotoistus on tärkeää, sillä eri valuutoilla on erilaiset arvot. Summa on virheellinen väärällä valuuttamerkillä. Jos valuutta olisikin oikeansuuruinen, sen muuntaminen lokalisoiduksi summaksi vie paljon aikaa. Ensinnäkin laskea se laskukoneella (tai tietokoneen ohjelmalla) ja toiseksi jos joutuu etsimään valuuttakurssia netistä tai sanomalehdistä.

Rahamäärien käsittelyssä on ongelmana myös eri alueiden erilaiset lainsäädännölliset määräykset. Tästä esimerkkinä ovat ALV[24] (arvonlisävero Suomessa) tai VAT[25] (value added tax Englannissa). Nykyisin kyseiset verot peritään molemmissa maissa, mutta ALV on tullut Suomeen vasta vuonna 1998. Tätä aiemmin suomalaisten käyttämissä sovelluksissa VAT:ille ei ole ollut vastinetta. Sovellukset voivat siis sisältää toiminnallisesti ominaisuuksia, jotka voivat peräti haitata sovelluksen käyttämistä kotoistettuja ohjelmia tai niiden käyttöä jossain toisessa paikassa

24 http://fi.wikipedia.org/wiki/Arvonlisävero (1.10.2007)

25 http://en.wikipedia.org/wiki/Value_added_tax (1.10.2007)

(kotoistamattominakin). Ongelman ratkeaa, jos sovelluksen lisätään käyttöpaikan selvittävä toiminnallisuus. Tämä voi johtaa kuitenkin monimutkaisiin päättelyihin ja lisää sovelluksen lähdekoodin kompleksisuutta.

Desimaalierottimen kotoistuksen tarve on erottaa se esim. tuhatlukujen erottimesta, joka on joissakin kotoistuksissa piste. Desimaalierotin tulee silloin olla pilkku. Jos tuhatlukuerotin on sananväli, silloin desimaalierottimeksi kelpaisi pistekin.

Kuten edellä tuli esille, tuhatlukujen erotin on käytössä joissakin tapauksissa, mutta ei aina.

Mitä ei saa kotoistaa?

Kaikkea ei kannata kotoistaa sovelluksissa. Tällaisia ovat mm. makrojen nimet. Esimerkiksi Microsoft Excel 97 funktioiden kotoistus teki englanninkielelle tehdyistä makroista toimimattomia. Edelleen ohjelmoijien oli vaikea opetella kotoistetut funktionimet kun he olivat tottuneet käyttämään alkuperäiskielisiä nimityksiä niille. Funktioiden nimet alkuperäisellä kielellä ovat yleisemminkin käytössä tietokone-ohjelmoinnissa, joten niistä poisoppiminen olisi ollut hankalaa.

WWW-sivuja voidaan lukea koneellisesti. Sivulta selataan avainsanoja sivun eri kohtia etsittäessä. Jos näitä avainsanoja kotoistetaan, etsintäskriptit eivät enää toimi sivulla. Koneellisia hakuja tekeviä voidaan tietenkin ohjata ja opastaa muuttamaan selaustoimintonsa kotoistetuilla sivuilla toimiviksi.

Dokumenttien kotoistus

Luvussa 3 käsittelimme avoimia dokumentteja. Myös niitä voidaan kotoistaa. Esimerkiksi tapahtumien ajat voidaan kotoistaa aikavyöhykkeiden mukaan. Näinollen dokumentin lukija ei erehdy esimerkiksi kokouksen alkamisajasta (kellonaika).

Asiakirjan merkistö, marginaalit, sivunumerointi yms. asiat ovat myös kotoistettavia asioita. On kuitenkin huomattava, että joissakin tapauksissa tiedon informaatio voi muuttua kotoistustoimenpiteen jälkeen. Esimerkkinä asiakirjastandardin mukaiseksi kotoistettuna sivunumerot muuttuvat ja tarkasteltava asia onkin eri sivulla.

Asiakirjojen kotoistuksessa tai yleensä siirrossa merkistöstä toiseen voidaan törmätä tilanteeseen, että kohdemerkistössä ei ole vastaavia merkkejä lähdemerkistön merkeille. Ongelman ratkaisemiseen ei ole olemassa yleispätevää käytäntöä. On olemassa transskriptioita, joiden mukaan jonkun merkistön merkki korvataan toisen merkistön merkeillä. (Korpela, 2005)

5.5 Kotoistustoimenpiteen ajankohta

Kotoistus on järkevintä aloittaa silloin kun sovelluksen kehittäminen jäädytetään ominaisuuksien suhteen. Tällainen ajankohta on betatestausvaihe. Silloin koodiin ei saa tehdä enää muita muutoksia kuin tärkeitä virheenkorjauksia. Uusia ominaisuuksia ei siis enää tule kuin erittäin pakottavissa tapauksissa. Ohjelman tekstit ovat tässä vaiheessa lopullisessa muodossa ja siten valmiita kotoistettaviksi.

Toki kotoistusta voi tehdä jo aiemmin, mutta sovellukseen voi tulla muutoksia ja kotoistus on tehtävä muutosten osalta uusiksi. Tehty kotoistustyö menee siis hukkaan.

5.6 Kotoistuksen toteuttaminen sovellukseen

Avoimien ohjelmien kotoistus toteutetaan ensin internationalisoimalla sovellus. Tämä muokkaa sovelluksen sellaiseksi, että kotoistus on siinä mahdollista. Käytännössä jokainen ohjelman tulostama teksti haetaan erillisestä resurssitiedostosta. Mitään ei ole kovakoodattu sovelluksen lähdekoodiin.

Kielelliset resurssit ovat avoimen ohjelmakoodin sovelluksissa *portable object template* (.pot -päätteinen mallitiedosto) tiedostona. Siitä ohjelman kääntäjät (translators) kotoistavat ohjelman haluamalleen kielelle. Kotoistettu tiedosto on *portable object* (.po -päätteinen) tiedosto. Yleensä tämä tiedosto tuodaan ohjelman kehitysympäristön palvelimelle versionhallinnan kautta. Portable object -tiedostot käännetään msgfmt-ohjelmalla *message object* (.mo -päätteiseksi binäärisiksi) tiedostoksi. Sovellukset osaavat käyttää näitä käännettyjä resursseja.

Kun sovelluksista kehitetään uusia versioita, sovellusten tekstuaaliset sisällöt muuttuvat. Ohjelmiin tulee uusia ominaisuuksia ja vanhoja olemassa olevia ominaisuuksia muutetaan. Tässä tapauksessa avoimen koodin ohjelmissa käännetyt tekstit muuttuvat sumeiksi (fuzzy). Siitä on ikävä seuraus: sovelluksen kotoistettu teksti korvautuu alkuperäisellä sovelluksen tekstillä. Tavallisesti tämä on englanti. Tämä näkyy sovelluksissa sekamelskana. On kotoistettua tekstiä ja alkuperäistä tekstiä esimeriksi yhdessä valikossa.

Kotoistajien tulisikin tarkistaa ja kääntää (translate) ohjelman muuttuneet tekstit uudelleen. Sumeiksi muuttuneiden tekstit ovat melkein samanlaisia aiempien versioiden kanssa. On vain pieniä muutoksia. Lisäksi monet tekstit muuttuvat sumeiksi, vaikka niissä mikään ei olisi muuttunut. Korjaustoimenpiteeksi riittää tekstin hyväksyminen sumeasta lopulliseksi käännetyksi.

Kotoistetut tekstit viedään versionhallinnalla varustettuun palvelimeen. Aika ajoin

sieltä kerätään ohjelmaan päivityspaketti. Päivitys koskee pelkästään sovelluksen kotoistettuja tekstejä. Päivitys tapahtuu automaattisella päivitysmekanismilla. Kehityksen innokkaat seuraajat voivat ladata palvelimelta uusia kieliversioita tiheämmin – esimerkiksi joka vuorokausi. He asentavat paketin järjestelmäänsä käsin.

Kotoistaja voi kääntää (translator) ohjelmia hyvin usein. Hänen kunnia-asianaan voi olla kotoistaa mahdollisimman uudetkin ohjelmaversiot. Silloin kotoistaja voi asentaa omalle kotikoneellensa versionhallintaohjelman. Se osaa hakea ohjelman palvelimelta yhdellä toimintokäskyllä kaikki edellisestä kerrasta muuttuneet ohjelmakoodit. Samoin hän voi yhdellä toimintakomennolla viedä palvelimelle tekemänsä muutokset. Versionhallintaohjelma pitää kirjaa eri kotoistajien sekä ohjelmistokehittäjien kontribuutioista, jotta ne eivät esimerkiksi menisi päällekkäin (ts. kaksi eri kotoistajaa kääntäisi saman ohjelmakohdan, eri tavoin).

Ohjelmien kotoistetut tekstit sijaitsevat sovelluksen käyttäjän tietokoneella textdomain -hakemistosta. Tämä on ilmaistu tietokoneessa ympäristömuuttujana (environment variable) TEXTDOMAINDIR. Jos ympäristömuuttujaa ei ole määritelty, kotoistettu teksti haetaan hakemistosta

/usr/lib/locale/lang/LC_MESSAGES,

jossa lang on kotoistuksen lokaali. Siinä on myös ilmaistu käytetty merkistö. Esim. suomenkielen lokaali UTF-8 -merkistöllä on

/usr/lib/locale/fi_FI.utf8/LC_MESSAGES.

Ohjelma gettext hakee kotoistetun tekstin tästä hakemistosta käyttäjän sovellukselle. Oikeammin on sanottava, että käyttäjän sovellus hakee kotoistetun tekstin hakemistosta gettextillä.

Käyttäjä voi lisätä tuohon hakemistoon ohjelmien kotoistettuja tekstejä. Ensin Internetistä haetaan halutun ohjelman .po -tiedosto. Sitten se käännetään se msgfmt:llä .mo -tiedostoksi. Nyt tiedosto on valmis vietäväksi em. hakemistoon. On muistettava asettaa tiedoston käyttöoikeudet. Yleensä tiedostot ovat vain pääkäyttäjän oikeuksin muutettavissa.

Matti Riikosen tutkimuksen mukaan java-ohjelmissa ohjelmien tekstit ovat resurssinipussa. (Riikonen, 2006) Näiden kotoistus poikkeaa edellä esitetystä tavasta. Lisätietoja viitteenä annetussa tutkimuksessa.

5.7 Julkistus

Kokoonpanon hallinnan (configuration management) yksi osa-alue on tuotejulkistusten hallinta. Ohjelman kehittäjien kannalta tuotteiden julkistuksen hallinnan päätehtävä on asettaa tuote saataville potentiaalisille käyttäjille yksinkertaisella tavalla, ja käyttäjien kannalta sen tulisi tarjota helppo tapa löytää ja hakea haluttu tuote.

Julkistusten hallintatarvetta lisää ohjelmistojen komponenttiluonne. Ohjelmistojen komponenttiluonnetta, komponenttien välisten suhteiden hajanaisuutta ja hajanaisuuden syitä sekä teoreettisia keinoja hajanaisuuden hallintaan tarkastellaan edellisessä luvussa versionhallinta.

Ennen vuosituhannen vaihtumista ohjelmistotuotannossa ja valmiiden ohjelmien julkistuksessa ilmeni ainakin kaksi ongelmakenttää (Hoek & al., 1997): Systeemien levitys oli vaivalloista, ja komponenttien paikallistaminen, hakeminen ja seuraaminen oli altista virheille. Tuotteiden osat oli jaettu erilaisten jakeluteiden kautta eri paikoissa ja erilaisin mekanismein. Eri osien riippuvuussuhteet olivat ja ovat monimutkaisia (Ousterhout, 1994). Ohjelmistojen dokumentointi oli perinteisesti jäänyt liian vähälle huomiolle. Lisäksi dokumentaatiot ovat olleet epätäydellisiä tai peräti puuttuneet kokonaan.

Näiden ongelmien ratkaisemiseksi kehitettiin julkistusautomaatti SRM. Sen periaatteita on sovellettu lähes kaikkiin tärkeisiin ohjelmistojen jakeluihin. Julkistuksen automatisointi on tullut mahdolliseksi käytännössä Internetin myötä.

Ohjelmien jakelu on siirtynyt 2000-luvun taitteessa Internetin kautta kulkevaksi. Internet sopii luonnollisen hyvin avointen ohjelmien jakeluväyläksi ollessaan sinänsä jo vapaa ja avoin vailla pidempiaikaisia jakelutien keskeytyksiä (kun verrataan perinteisiä logistiikkaväyliä). Nykyisin (vuonna 2007) suurin osa yleisesti käytettyjen sovellusten jakeluista ja päivityksistä tapahtuu Internetin välityksellä.

KDE:n julkistusten hallintaan (KDE Release Management) kuuluu aikataulutus, uusien ominaisuuksien ja virheenkorjausten listaus, kielenkäännökset, distribuutioiden hallinta ja tiedotus.

5.8 Ohjelmien kehittämisen edistyminen

Forrester Research arvioi vuonna 2000 tutkimuksiinsa perustuen, että ohjelmistotuotannon standardit muuttavat koko ohjelmistoteollisuuden vuoteen 2004

mennessä[26]. Aivan näin rajuihin muutoksiin avoin lähdekoodi ei ole johtanut. Kuitenkin tutkimuksessa esitettyjä piirteitä on nähtävissä. Turvallisuuden lisäystarpeet ovat yksi tekijä muutostarpeisiin. Internet on ratkaisevasti lisännyt tietoturvauhkia ja toiseksi maapallon globalisaatiosta johtuva uusien alueiden tulo omine ominaispiirteineen.

Vapaiden ei kaupallisten sovellusten kehitystyö eteneminen on epämääräisempää kuin kaupallisten vastaavien kehitys. Ensinnäkin suurinta osaa sovelluksen kehittäjiä ei motivoi rahallinen hyöty. Toiseksi hän tekee sitä silloin kun saa idean uudesta ominaisuudesta ja on aikaa toteuttaa se tai intoa korjata virhe tai hän on keksinyt tavan korjata sen. Esimerkkinä tällaisesta ovat KDE:n julkistusaikataulu[27], jossa sanotaan, että julkistus riippuu julkistajan voinnista Dublinin matkan jälkeen.

Toinen esimerkki on graafinen ohjelmisto Gimp (Gimp, 2008), jonka kehittäminen oli pysähdyksissä vuosikausia. Sitten siihen tulee lyhyen ajan sisällä monta uutta versiota uusine ominaisuuksineen.

Linuxin distribuutioiden lukumäärä on lisääntynyt muutaman vuoden aikana voimakkaasti. Niitä on nykyään kymmeniä. Ohjelmien yhteensopivuuden kannalta lukumäärä olisi hyvä pysyä kohtuullisissa rajoissa. Onko distribuutioita tehty miettimättä tarpeeksi tarpeita. Linuxin ytimen epäyhteensopivuudet tulevat näkyviin moduulien puuttumisena. Käyttäjän on etsittävä laitteistoaan vastaava moduuli ja asennettava se käyttöjärjestelmän ytimeen. Joskus joutuu jopa kääntämään käyttöjärjestelmän ytimen (kernel) uudelleen.

26 http://sektori.com/uutiset/1613/forrester Forrester Research: avoin lähdekoodi muuttaa koko ohjelmistoteollisuuden (10.3.2007)

27 http://developer.kde.org/development-versions/kde-3.5-release-plan.html (22.9.2006)

6 Avoimen lähdekoodin ohjelmien ylläpito

Ohjelmien elinkaareen kuuluu kehittelyn ja julkaisun jälkeen ylläpito. Sen tarkoitus on säilyttää ohjelmisto toimivana sekä estää ja korjata vikoja. (Koponen, 2007) On olemassa useita ylläpitomalleja, jotka kukin korostavat jotain ylläpidon tehtävää. Ylläpito on toimenpiteinä ohjelmien asennuksia, asetusten säätämisiä, virheiden toteamisia sekä niiden mahdollista raportointia ja ohjelmien poistamisia.

Avoimen lähdekoodin ohjelmien ylläpito on melko samanlainen prosessina kuin ISO/IEC ylläpitomalli. ISO/IEC 14764 -ylläpitomallissa ylläpito koostuu ensin ohjelmiston asennusprosessista ja sen jälkeen syklisesti virheen havaitsemisen, muutoksen ja muutoksen käyttöönoton sarjana lopulta poistoprosessiin tai liittämiseen toisiin ohjelmistoihin. Avoimen lähdekoodin ylläpito on kuitenkin hajanaisempaa eikä yhtä kurinalaita. (Koponen, 2007)

Avoimen lähdekoodin ohjelmien ylläpidossa on löydettävissä ainakin kaksi järjestelmää. virheidenhavaitsemisjärjestelmä (DMS) ja versionhallintajärjestelmä (VMS). (Koponen, 2007)

Ylläpito sisältää 1) käyttäjien löytämien virheiden raportointi DMS-järjestelmään, 2) kehittäjät hakevat raportteja DMS-järjestelmästä analysoiden niitä, 3) kehittäjät korjaavat havaitut virheet lähdekoodissa. Korjattu koodi tarkistetaan ennen hyväksymistä ja sen jälkeen se viedään VMS-järjestelmään. Sen jälkeen DMS-järjestelmässä muutetaan virheraportin tila korjatuksi. Myöhemmin ohjelman käyttäjä voi hakea uuden korjatun ohjelmaversion VMS-järjestelmästä. (Koponen, 2007) Korjattu versio voi johtaa myös ohjelman versionumeroinnin muutokseen.

6.1 Virheiden ilmoitus

Virheetöntä tietokoneohjelmaa ei ole olemassa. Ohjelmistojen kehitysympäristöön kuuluu myös virheiden ilmoitusjärjestelmä. Virheidenilmoitusjärjestelmät ovat olleet perinteisesti asiakas-palvelin -tyyppisiä ohjelmia. Nykyisin ne ovat www-pohjaisia, eivätkä ne tarvitse erityisiä asennettavia ohjelmia käyttäjän koneelle. WWW-pohjaisina niitä voi käyttää suoraan ympäri maapallon sijainnista riippumatta – juuri kuten tehokas virheidenilmoituksen tuleekin toimia. Järjestelmän tulee toimia useilla selaimilla.

Suurin osa saatavilla olevista kaikista järjestelmistä on Microsoft Windows -pohjaisia. Ne tarvitsevat siksi NT/2000/XP -palvelimen ja SQL Server -tietokannan. Suurin osa Linuxissa toimivista virheidenilmoitusjärjestelmistä ovat vapaita. Niiden

asentaminen ja käyttö vaatii enemmän teknisiä taitoja. On olemassa vain harvoja samalla lähdekoodilla toimivia useamman alustan virheidenkorjausjärjestelmiä. Jotkut tällaiseksi väitetyt järjestelmät ovat kuitenkin aivan omia tuotteita kullekin järjestelmälle (http://www.websina.com/bugzero/bug-defect-tracking.html).

Useimmissa virheidenkorjausjärjestelmissä on tietokanta tietojen talletukseen. Harvat tallentavat tiedot tiedostoiksi. Tietokantajärjestelminä niissä on yleisimmin Microsoft Access tai MySQL. Vain harvoissa järjestelmissä tietokanta voidaan valita. Järjestelmässä voi olla versionhallinta kuten CVS (CVS, 2000). Se mahdollistaa ohjelmakoodin käsittelyn virheidenilmoitusten yhteydessä.

Monista virheidenkorjausjärjestelmistä ei ole saatavissa lokalisoituja versioita. Lokalisoituja kohteita niissä ovat www-käyttöliittymä, tiedot, ja sähköpostitse lähetettävät ilmoitukset. Jos tarvitaan lokalisoitua järjestelmää, sen tulisi olla kompleksisen tehtäväkentän takia helppokäyttöinen.

Virheidenilmoitusjärjestelmät on koodattu c:llä/c++:lla, perlillä/php:llä, tai javalla. Käytetty kieli kannattaa valita sen mukaan, millaiseen ympäristöön järjestelmä tulee. Esimerkiksi java-ohjelmien virheidenkorjausjärjestelmän luonnollinen toteutuskieli on java.

Avoimien ohjelmien maailman virheidenilmoitusjärjestelmissä on myös virheiden kommentointimahdollisuus, jotta toiset virheistä kiinnostuneet aktiivit käyttäjät voivat antaa lisätietoja virheestä tai antaa neuvoja, miten virhetilannetta ei synny tai jopa paikan (patch) virheelle. Ilmoitusjärjestelmissä on usein myös pisteytys virheille ja käyttäjien äänestysmahdollisuus. Tällä tärkeät, käyttäjiä erityisesti haittaavat virheet tulevat paremmin esille.

Järjestelmissä on ilmoitustoiminto, jolla ohjelman käyttäjät voivat ilmoittaa ohjelman virheestä. Käyttäjän koneella ohjelmassa voi olla automaattinen ilmoitusjärjestelmä. Tämä tarkoittaa sitä, että kun ohjelmassa tapahtuu virhe, se ilmoittaa siitä automaattisesti virheidenilmoitusjärjestelmään ilman käyttäjän tekemiä toimenpiteitä. Jos koko ohjelma kaatuu, järjestelmässä oleva tunnistusohjelma tunnistaa tilanteen ja raportoi siitä virheidenkorjausjärjestelmään. Käyttäjällä tulee olla kuitenkin mahdollisuus estää automaattiset ilmoitukset niin halutessaan yksityisyyden suojan takia.

Ilmoitusjärjestelmissä on myös mahdollisuus antaa toiveita uusista ominaisuuksista ohjelmiin. Toiveissa kuten myös tavallisissa virheraporteissa on pisteytysjärjestelmä. Tärkeää toivetta voi äänestää ja näin se saa enemmän painoarvoa. Koodaajat asettavat paljon ääniä saaneet toiveet etusijalle ja korjaavat paljon ääniä saaneita virheitä muita virheitä nopeammin.

Tunnettu avoimien ohjelmistojen virheidenilmoitusjärjestelmä on Bugzilla. (Bugzilla, 1998) Virheidenilmoitusjärjestelmiä on kehitetty eri ohjelmointikielillä Javalla + J2EE:llä Itracking, C#:lla ASP.NET -tekniikkaa käyttävä BugBye.

Avoimien koodin ohjelmistoihin voisi periaatteessa avoimuuden syystä syntyä paljon haittaohjelmia, sillä kuka tahansa kykenevä pystyisi sellaisen luomaan. Kuitenkin asia on päinvastoin. Avoimiin ohjelmistoihin ilmestyy virheenkorjauspaikkoja nopeammin kuin kaupallisiin suljetun koodin ohjelmistoihin. Paikkojen jakelu on myös nopeampaa. Verrataanpa vaikkapa Microsoftin kerran kuukaudessa (tai harvemmin) tuleviin ns. service packeihin.

Avoimissa ympäristössä virheenkorjaus voi tulla nopeasti. Esimerkiksi KDE-työpöytäohjelmiston virheenilmoitusjärjestelmän viikkoraporttien[28] mukaan viikolla 38/2006 nopein virheenkorjaus on tullut 9 minuutissa ilmoituksen jälkeen. Avoimen koodin ohjelman käyttäjä voi samantien ladata lähdekoodin palvelimelta ja kääntää omalla koneella ohjelmasta uudemman version, jossa ei kyseistä virhettä enää ole. Jos hän malttaa odottaa seuraavaan päivään, hän voi ladata palvelimelta suorituskelpoisen ohjelman uuden version (snapshot).

28 http://bugs.kde.org/weekly-bug-summary.cgi Top 20 people who most quickly fixed a report in the last 7 days (24.9.2006)

7 Esimerkkejä avoimista teknologioista

Avointen ohjelmistojen kehitysympäristöjen tyypillisiä piirteitä on rajoittamaton julkinen näkyvyys, versionhallintajärjestelmä, virheidenilmoitus. Kuka tahansa voi liittyä ja ottaa osaa kehitystyöhön. Avoimen koodin leviäminen erilaisiin sovelluskohteisiin on alkanut voimakkaasti vuoden 2005 vaiheilla.

Terveydenhoidossa röntgenkuvat ja muut kuvat siirretään ja talletetaan digitaalisesti. Niitä käytetään diagnoosien apuna tietoverkkojen kautta. Toiminta on nopeuttanut lääkärien diagnoosien tekoa. Yksi tällaiseen tarkoitukseen käytetty järjestelmä ja ohjelmisto on PACS. Siitä on olemassa myös avoimen lähdekoodin ohjelma[29]. Terveydenhoidon sovellusalueella ohjelmilta vaaditaan tavallista enemmän. Tietoturvan on oltava hyvä ja potilasturvallisuuden takia niiden on toimittava virheettömämmin kuin tavallisessa toimistokäytössä.

Hallinnollisissa laitoksissa tulevat ja lähtevät asiakirjat luetteloidaan diaarijärjestelmään. Diaarijärjestelmässä on päivämäärätiedot, lähettäjä ja vastaanottaja sekä muita käsittelyssä käytettyjä merkintöjä. Hyvä hallinnointitapa edellyttää diaarijärjestelmän ylläpitoa.

Geneven autonäyttelyssä 2007 (8. - 18.3.2007) BMW ja Intel julkistavat proto-tyyppinä viihdejärjestelmään integroitua läsnäolo-, paikannus- ja videopuhelu-teknologiat avoimella koodilla toteutettuna[30]. Tämä on ensimmäinen kerta, kun sarjarakenteiseen autoon sijoitetaan avoimen lähdekoodin ohjelmia.

Avointa koodia pyritään levittämään myös avaruusteknologiaan. NASA on tutkinut avoimen lähdekoodin kehitysmallia (NASA, 1989) ja harkitsee sen käyttöönottoa. Niistä olisi tutkimusten mukaan hyötyä NASA:lle. Avoimen ohjelmistojen eduksi lasketaan helpon arvioinnin ja riippumattomuuden ohjelmantuottajasta. Riippumattomuus varmistaa ohjelman käytettävyyden sen koko elinkaaren aikana . Käytön esteinä on lähinnä Yhdysvaltojen vientirajoitukset.

Seuraavassa esitellään kaksi tunnettua avointen ohjelmistojen kehitysympäristöä. Lisäksi tutustutaan kolmeen avoimen lähdekoodin ohjelmistoon.

29 http://www.mii.ucla.edu/index.php/MainSite:OpenSourcePacsHome Open Source PACS Home (24.1.2007)

30 http://www.digitoday.fi/page.php?page_id=9&news_id=20075909 Movial sulloo Bemariin avointa lähdekoodia (8.3.2007)

7.1 COSS

Suomen avointen ohjelmistojen keskus COSS kokoaa yhteen suomalaisia avoimien ohjelmien kehittäjiä, niiden käyttäjiä, kerää tietämystä ja edistää avointen ohjelmien leviämistä Suomessa. Se pyrkii kehittämään avoimen lähdekoodin ohjelmia liiketoimintaympäristöön. (COSS, 2003)

COSS yhdistää avoimen lähdekoodin ratkaisuja ja palveluja tarjoavia yrityksiä ja yhteisöjä niitä ostavien tahojen kesken. Toiminta on kansallista ja kansainvälistä. Palveluihin kuuluvat tietopalvelut, koulutus ja konsultointi mm. lisensiointiin ja tekijänoikeuksiin liittyvissä kysymyksissä. COSSin tarkoitus on levittää tietoutta avoimista ratkaisuista. Coss aktivoi ja koordinoi tutkimusta yhteistyössä suomalaisen tutkimus- ja kehitysprojektien rahoittajan Tekesin sekä johtavien yliopistojen kanssa. Tapahtumat ja seminaarit ovat myös COSSin toimintamuoto. Toiminta on monipuolistunut paljon noin viiden vuoden aikana tämän tutkimuksen kirjoitus-hetkellä.

Filosi (Finnish Linux and Open Source Initiative) on foorumi, joka ideoi ja tukee innovatiivisia tutkimushankkeita. Filosi on poikkitieteellinen hanke. Siinä on mukana yliopistoja ja tutkimuslaitoksia. Ossi-tutkimusprojektissa luodaan työkaluja yrityksille open source -operaatioiden johtamiseen julkaisujen ja raporttien muodossa. ServOSS-tutkimushankkeessa pyritään edistämään suomalaisten avoimia ratkaisuja käyttävien yritysten kansainvälistymistä.

Avoimien ohjelmien koodin kehitystä edistetään järjestämällä tapahtumia ja yhteisöjen sisäisiä tapaamisia. COSS organisoi kesäkoodi-nimisiä projekteja opiskelijoille kesätöiksi. Projektissa toteutetaan ohjelma tai menetelmä. Kesäkoodi-toiminta on alkanut lupaavasti. Avoimen lähdekoodin teknologioita seurataan ja COSS luo ja koordinoi yhteisprojekteja yritysten, kehittäjien ja julkisen sektorin välille.

7.2 SourceForge

SourceForge[31] on maailman laajin avointen ohjelmien kehitysympäristö. Ympäristö tarjoaa käyttäjille projektinhallinnan, ohjelmiston versionhallinnan, virheiden-ilmoitusjärjestelmän, keskustelualueen ja ohjelmien latauksen. Sen omistaa OSDN (Open Source Development Network), jota sponsoroi VA Software Corp (Nasdaqissa LNUX). SourceForge on vapaa. Sillä on (tilanne v. 2007) yli miljoona käyttäjää ja siinä on yli 100 000 projektia. Järjestelmässä on laajin saatavilla oleva kokoelma

31 http://www.sourceforge.net/ SourceForge (13.1.2007)

ohjelmien lähdekoodeja ja käännettyjä ohjelmia. (SourceForge, 2004)

OSDN:n juuret ovat peräisin Andover.net -yhteisöstä (OSTG, 2001). Vuonna 1996 syntyi ajatus tarjota ennakkoluulottoman ja rajoittamattoman sisällön, yhteisön ja kauppaympäristön palvelu Linuxin ja avointen ohjelmien yhteisöille. VA Software lisäsi sponsoroinnin kohteeksi Slashdotin ja Freshmeat.net:in teknologiaryhmäänsä sekä ThinkGeekin ja AnimationFactory.com:in elektroniseen liiketoimintaansa. Näin SourceForge sai lisää merkittävyyttä ja suosittavuutta.

7.3 Linuxin kehitysympäristö

Avoimien ohjelmistojen paras esimerkki on Linux-käyttöjärjestelmä. Ensimmäinen virallinen versio julkistettiin 30.4.1994. Linux on hioutunut 15 vuodessa teknisesti korkealaatuiseksi. Tekniikan hyvyys ei ole kuitenkaan tärkein puoli Linuxin hyvyydessä. Kaikkein eniten Linuxin kehitykseen ja leviämiseen on vaikuttanut yhteistyö. Linux ei ole ollut ainoastaan vallankumouksellinen vapaana käyttöjärjestelmänä, vaan se loi uuden kehitysympäristön.

Linux oli alunperin Helsingin yliopiston opiskelijan Linus Torvaldsin töiden ajoitus (skedulointi) -aiheinen harjoitustyö, jota hän kehitteli edelleen paremmaksi varsinaisen harjoitustyön päätyttyä. Sekään ei ollut merkittävää Linuxin voittokululle. Linus julkaisi työnsä julkisesti Internetissä ja pyysi kommentteja siihen. Korjausehdotuksia tulikin paljon ja Linus toteutti korjaukset. Sen jälkeen hän julkaisi korjatun version Internetissä. Tämä on ollut ehkä merkittävin tekijä Linuxin voittokulun alkamiseen. Tässä vaiheessa se ei ollut ollenkaan vakavasti otettava käyttöjärjestelmä. Vain nörttien kokeilu käyttöjärjestelmäksi.

Kehitystyö laajeni entisestään ja jossain vaiheessa Linus siirtyi koodin kehityksestä enemmän hallinnollisiin tehtäviin. Linuxin kehitystyö hajautui. Linus piti vain kehityksen ohjakset käsissään.

Linux-käyttöjärjestelmän leviämiseen on vaikuttanut asema valta-asemassa olevan Microsoft Windowsin haastajana. Ennen Linuxin tuloa edes minkäänlaiseksi kilpailijaksi Microsoft Windowsille, monet eivät pitäneet Microsoftin monopolimaista markkinaosuutta hyvänä asiana. Alkuvaiheessa haastajalle tulvii runsaasti sympatiaa ja sen kasvamista kilpailijaksi jopa tuetaan. Vuonna 2006 Microsoftin markkinointi on ryhtynyt ponnekkaasti vastustamaan Linuxia. Tämä on merkki todellisen kilpailutilanteen alkamisesta. Ylläpitokustannusten edullisuustutkimusten lisäksi Microsoft on solminut lisenssisopimuksia Novellin kanssa. Julkisuudessa on keskusteltu motivaatioita tälle kaupalle. Joidenkin arvioiden mukaan Microsoft haluaisi rahastaa lisenssimaksuja Linux-järjestelmän käyttäjiltäkin. Toisten arvioiden

mukaan Microsoft haluaisi patenteilla estää muut (lähinnä avointen ohjelmistojen) Linuxin tuotekehitykset.

MS Windows NT suunniteltiin aivan uudelta pohjalta, olemaan monen käyttäjän järjestelmä. Siitä lähtien XP ja Vista ovat parantaneet NT:ssä havaittuja huonoja ominaisuuksia. Kuitenkin Windowsissa on perustavaa laatua olevia suunnittelu-periaatteita, jotka soveltuvat huonommin monen käyttäjän järjestelmäksi.

Linuxille on ollut etuna olla vähän konetehoa vaativana ohjelmistona (ja sen päällä olevat käyttöliittymäohjelmat, kuten KDE, Gnome, XFCE). Se on tullut median kautta julkiseen tietoisuuteen. Nyttemmin ero on pienentynyt kun Linux-ohjelmiin on tullut lisää käyttöä helpottavia ja nimenomaan graafisia ominaisuuksia. Tämä on pienentänyt Linuxin vähäisten resurssitarpeiden etumatkaa MS Windowsiin nähden.

On mielenkiintoinen piirre, että Windowsin virheistä ja huonosta toiminnasta syytetään Microsoftia. Vastaavanlaiset tilanteet avointen lähdekoodien ohjelmien käytössä ovat käyttäjän omaa tyhmyyttä. Linux-järjestelmän käyttäjä kuulee usein toteamuksen, että se on korjattava itse. Kirjoittaja on havainnut myös, että tietokoneiden ja ohjelmien käyttäjät tyytyvät siihen, jos he voivat tehdä virheilmoituksia ja antaa kehitysehdotuksia ohjelmista. Silloin virheistä ei niin paljon moitita ja virheitä yritetään kiertää.

Linux-distribuutioista Ubuntu on nousemassa suosituimmaksi käyttöjärjestelmäksi. Suosion syynä voivat olla palvelut, joita järjestelmän mukana tarjotaan. Käyttäjätukea saa keskustelufoorumeissa. Ubuntun hyvät mediaominaisuudet ja ohjelmien helpot asennukset pakettivarastoineen ovat edesauttamassa Ubuntun leviämistä.

7.4 KDE

KDE eli K Desktop Environment on alettu kehittää julkisesti noin vuonna 1998. Se on ollut ensimmäisiä graafisia käyttöliittymiä PC-koneissa toimivalle Unix-tyyppiselle Linux-järjestelmälle. Tällä hetkellä vuonna 2006 se on kehittynein, käytetyin ja laajalle levinnein Linux-käyttöliittymä. KDE-työpöytäohjelmisto on käännetty 65 eri kielelle. (KDE, 1998)

KDE-ohjelmat perustuvat norjalaisen TrollTech yhtiön Qt-ohjelmointirajapinnalle[32]. Trolltech lisensioi sen GPL-lisenssin alaiseksi versiosta 2.2 lähtien lokakuussa vuonna 2000. Ohjelmankehittäjät eivät hyväksyneet aiempia QPL-lisenssiehtoja[33].

32 http://trolltech.com/company/customers/coolapps Qt Applications – Trolltech (14.6.2008)

33 http://www.linuxdevices.com/news/NS4030623636.html Qt under GPL (14.6.2008)

KDE-järjestelmä on kokoelma tälle alustalle kehitettyjä ohjelmia. Uusiin KDE:n versioihin otetaan aika-ajoin uusia hyviksi havaittuja KDE-ohjelmia. Ohjelmia löytyy koulumaailmaan, Internetin hallintaan, sähköposteihin, uutisiin (news), RSS-syötteisiin, kalenteriohjelmia, yms. Kokoelmasta voi tippua pois huonolaatuinen ohjelma. Näin on käymässä esimerkiksi sähköpostiohjelmalle Kmail sen sisältämien ohjelmavirheiden takia KDE:n versiossa 4.0.

KDE:n kotisivu sisältää ryppään palvelinkoneita. Palvelin www.kde-apps.com sisältää KDE:lle tehtyjä ohjelmia. Palvelin www.kde-files.org sisältää KDE-ohjelmille tehtyjä mallipohjia ja resursseja. Palvelin www.kde-look.org sisältää ulkonäköön liittyviä paketteja kuten taustakuvia, teemoja, kuvakkeita, ääniä jne. Kullakin palvelimella tuotuja resursseja voi etsiä uutuuden tai pisteytyksen mukaan. Palvelimella bugs.kde.org on virheidenilmoitusjärjestelmä.

KDE:n käännökset eivät ole täydellisiä, vaan esimerkiksi suomennustaso on noin 80%. Tämä näkyy ohjelmissa siten, että osa teksteistä on englanninkielisiä. MikroPC-lehti onkin kritisoinut valikkojen kaksikielisyyden sekavuutta. Lisäksi elektroniset ohjelmakäsikirjoja ei ole suomennettu. (MikroPC, 2000) Tutkielman kirjoittaja on kääntänyt KDE-ohjelmia suomeksi ja on pyrkinyt kääntämään näkyvät osat ohjelmista suomeksi. Silloin ohjelman käyttäminen on miellyttävämpää. Tilanne on muutenkin muuttunut oleellisesti uusien versioiden mukana kun aikaa on kulunut tuosta hetkestä 7 vuotta.

7.5 OpenOfficeOrg

StarOffice suiten tekijäyhtiö StarDivision perustettiin Saksassa 1990-luvun puolivälissä. StarOfficen versio 5.2 julkistettiin kesäkuussa 2000. Sitä aiemmin Sun Microsystems osti StarDivisionin kesällä 1999. Myöhemmät StarOfficen versiot versiosta 6.0 on kehitelty OpenOffice.org:in lähdekoodeista, API-rajapinnoista, tiedostomuodoista ja referenssitoteutuksista. (OpenOfficeOrg, 1999).

OpenOffice.org:in ohjelmaperheeseen kuuluu tekstinkäsittely, taulukkolaskenta ja tietokantaohjelma. Siitä on saatavana myös SDK (Software Development Kit) ja ajonaikainen ympäristö (runtime environment). Ohessa sen mukana jaetaan myös sanakirjoja eri kielille tavutuksiin ja oikeinkirjoituksen tarkistuksiin, clipart-kuvia ja mallipohjia.

OpenOffice.org käyttää LPGL-lisenssipolitiikkaa. OpenOffice.orgin suomennos-ryhmä palkittiin vuonna 2006 käännöksestä.

Tekstinkäsittelyohjelman käyttökelpoisuutta Suomessa parantaa suomenkielen

tavutusohjelma. Avointen tekstinkäsittelyohjelmien ja muissakin avoimissa ohjelmistoissa suomenkielen tarkistusohjelmana on ollut ilmainen, mutta suljetun koodin Soikko. Tämä nähtiin huonona avoimien ohjelmien käytön yhteydessä. Lisäksi sen kehitys ja tulevaisuus on suljettuna koodina epävarmaa. Noin vuonna 2004 käynnistyi avoin sanojen tunnistusohjelma Voikko (Voikko, 2006). Sen kehittelyä on tukenut COSS. Voikko tunnistaa suomenkielisiä sanoja taivutuksineen. Tunnistus tapahtuu jo varsin hyvin. Sanastoja laajennetaan koko ajan julkisella palvelulla, jonne jokainen voi lisätä itse uusia sanoja. (Joukahainen, 2006)

8 Avoin teknologia julkisessa hallinnossa

Tässä luvussa perehdymme avoimiin teknologioihin hallinnossa kahdesta näkökulmasta. Ensin selvitämme julkishallinnon avoimien teknologioiden projekteja. Toiseksi tutkimme avoimien teknologioiden levinneisyyttä ja käyttöä julkishallinnossa, koulumaailmassa ja muissa julkisissa palveluissa. Tutkimme myös ODF:n asemaa julkishallinnossa.

Julkisessa hallinnossa käsitellään ja tallennetaan suuria tietomääriä. Tietojen oikeellisuus ja tietoturva ovat tärkeitä. Tiedoilla on suuri merkitys ja ne ovat sidoksissa toisiinsa monella tavalla. Ajatellaanpa henkilön verotietoja, potilastietoja ja eri ministeriöiden tietojenkäsittelyä.

Julkisen hallinnon toimistoissa käsitellään ja tuotetaan runsaasti tietomassaa. Ennen tietokoneiden aikaa tietoja on talletettu paperisessa muodossa ja mikrofilmeillä. Tietojen käsittely on siirtynyt tietokonepohjaiseksi Microsoft Windowsin tultua markkinoille vuodesta 1993 lähtien. Windows Office on ollut ensimmäinen ohjelmisto, jossa on toimiston perustoimet kuten tekstinkäsittely, laskenta, tiedon talletus ja organisointi (tietokanta) sekä tiedon välitys kuten sähköpostit ja faksit. Lähes kaikki tietojenkäsittelytoiminnot, joita toimistoissa suoritetaan.

Yksittäisellä työasemalla tietojen välitykseen aiemmin mainittujen ohjelmien välillä on käytetty OLE-tekniikkaa (Object linking and embedding)[34]. Tallennetut tiedostot on voitu avata Micrsoft Windows -ohjelmaperheen ohjelmilla. Muiden ohjelmien kanssa on ollut vaikeuksia tai tietojen siirto ei ole ollut ollenkaan mahdollista.

Yksi tietojen käsittelyn tehostumisen seuraus on tiedon tuottavuuden kasvu. Se on lisännyt tuotetun tiedon määrää. Automatisoinnin hyöty ja vaikutus on nimenomaan käsiteltävien tietomäärien kasvu.

Julkinen hallinto kuten myös elinkeinoelämä on ollut tiiviisti Microsoftin tuotteiden käyttäjiä toimisto-ohjelmien suhteen aina 1990-luvulta asti Microsoft Windowsin markkinoille tulon jälkeen 2000-luvun alkupuolelle. Microsoft on ollut ylivoimaisesti paras ja monipuolisin toimistokäyttöön tarkoitettujen ohjelmien valmistaja. Yksittäisissä tehtävissä jonkun muun valmistajan ohjelma on ollut se de facto kun Microsoftin vastaava tuote on ollut ominaisuuksiltaan tai käytettävyydeltään huonompi tai vastaavaa tuotetta ei ole ollut ollenkaan. Todellisia kilpailijoita toimisto-ohjelmien kokonaisuudelle ei ole ollut.

Hallitseva markkina-asema on mahdollistanut sanelupolitiikan standardeihin. Tätä on

34 http://searchwinit.techtarget.com/sDefinition/0,,sid1_gci214126,00.html (8.10.2007)

pidetty huonona asiana. Sanelupolitiikalla valmistaja voi suunnata käyttöä haluamallaan tavalla. Esimerkiksi sellaisilla, joka lisää valmistajan saamia rahallisia tuottoja. Tai sellaisella, joka estää kilpailevien tuotteiden käyttämisen yhteensopimattomilla protokollilla tai lisenssiehdoilla. Edelleen hallitsevana markkinoiden hallitsevana valmistajana joku yhtiö voi määrätä kehityksen suunnan. Se voi olla kokonaisuuden kannalta epäedullinen esimerkiksi rajoittamalla tuotteen käyttökohteita.

Microsoftin lisäksi on muitakin määräävässä asemassa olevia yhtiöitä. Esimerkiksi Sun Microsystems java-teknologioineen on merkittävä suunnan näyttäjä avointen lähdekoodienkin alueella.

8.1 Julkishallinnon projekteja

Vuoden 2005 alkupuolella Eric Kriss, Secretary of Administration and Finance Massachusettsissa, oli ensimmäinen osavaltion hallinnon virallinen edustaja Yhdysvalloissa joka yhdisti avoimet tiedostomuodot julkiseen käytäntöön tarkoituksella: "It is an overriding imperative of the American democratic system that we cannot have our public documents locked up in some kind of proprietary format, perhaps unreadable in the future, or subject to a proprietary system license that restricts access." At a September 16, 2005 "New England Town Meeting."

Käännös kuuluu vapaasti suomennettuna: "On ylitsepääsemättömän tärkeätä amerikkalaiselle demokratialle saada tiedostojamme säilöttyä jonkinlaisella sovelluksesta riippumattomalla tiedostomuodolla, koska muuten tulevaisuudessa tiedostomme ovat ehkä lukemattomissa tai alttiita ohjelmistokohtaisille lisensseille jotka rajoittavat käyttöä." Kriss korosti, että hän uskoo tämän olevan perustavaa laatua oleva suvereeni kysymys. Hän tukee yhtiöiden oikeuksia saada ideaalinen omistusoikeus, mutta yhdenkään yhtiön ei tulisi saada hallita sitä mitä osavaltio tekee osavaltion omille tiedoille.

Vuoden 2000 alkupuolella hallinnollisella alalla syntyi näkyvää kiinnostusta avoimiin ohjelmiin. Tehtiin tutkimuksia avoimien ohjelmien käyttökelpoisuudesta. Koska yksi avoimien tiedostomuotojen kuten OpenDocument tavoitteista on varmistaa pitkäaikainen pääsy tiedoston sisältöön ilman teknisiä tai juridisia esteitä, ovat hallitukset tulleet yhä tietoisemmiksi avoimien tiedostomuotojen eduista.

Vuonna 2002 tohtori Edgar David Villanueva Nuñes, lakimies ja Perun tasavallan kongressin edustaja, kirjoitti kirjeen Microsoft Perulle nostaen esiin kysymyksen vapaasta ja pysyvästä asiakirjan sisällöstä sovelluskohtaisilla muotoiluilla.

Euroopassa

Eurooppalaisella tasolla julkishallinnollisten ohjelmistojen ja standardien koordinoija on IDABC[35]. European Internet Foundation EIF:n[36] päämäärä on tarjota kehityksen johtajuus poliittisille, taloudellisille ja sosiaalisille haasteille digitaalisessa mullistuksessa. Nämä ovat luoneet määritelmän avoimuuden standardille Euroopassa.

Information Society Technology (IST) tutkii ja analysoi vapaan ohjelmiston toimialaa ja tekee suosituksia. Vuonna 2002 julkaistiin IST:n FLOSS-projektin raportti. Raportissa esitellään vapaiden ohjelmistojen historiaa, leviämistä ja kaupallisia vaikutuksia. Interchange on Data between Admministrators tutkii vapaita ja avoimia ohjelmia julkisella sektorilla. (Aalto-Setälä & Leppäniemi, 2005)

EU suosittelee avoimen lähdekoodin järjestelmiä. Suosituksen päämääränä on luoda yhtenäinen ympäristö alueella käytettäville ohjelmille. Tavoitteena on luoda sähköisten palvelujen tarjoaminen kansalaisille. On kuitenkin esitetty varoituksia, että avoin lähdekoodi on eri asia kuin avoin järjestelmä. (Sektori, 2003)

EU rakentaa yhteisen avoimen lähdekoodien varaston ja tietopankin nimeltään Osor (Open Source Observatory and Repository). Siellä on ohjelmakoodien ja valmiiden ratkaisujen lisäksi tietoa ohjelmien käytöstä, sopimuspohjia ja lisenssitietoa. (OSOR, 2005)

Euroopan Unionin Komissio suosittelee OpenDocument-tiedostomuodon käyttöä jotta hallinnon asiakirjojen käsittely ei olisi riippuvaista yksittäisen ohjelmisto-valmistajan toiminnasta. Sen lisäksi avion standardi on käyttökelpoinen pidemmän ajan. Sitähän hallinnollisilta dokumenteilta vaaditaan.

Suomessa

Suomessa ensimmäisiä avoimen lähdekoodin ohjelmia tutkinut julkinen taho oli Turun kaupunki. Tutkimus lähti ajatuksesta, että avoimet ohjelmat ovat ilmaisia, niillä voidaan korvata valalla oleva kaupallinen vaihtoehto ja näin ollen saavutetaan säästöjä. Tutkimukselle asetettiin etukäteen odotuksia, jotka eivät toteutuneet. Loppujenlopuksi tästä syystä tutkimus päättyi lopulta katkerasti riitoihin. (Aalto-Setälä & Leppäniemi, 2005) Tutkimuksen ansioksi laskettakoon kuitenkin, että se oli

35 http://europa.eu.int/idabc/ IDABC:n kotisivu (4.10.2007)

36 http://www.eifonline.org/ EIF kotisivu (4.10.2007)

ensimmäisiä julkisia avointen ohjelmien käytön selvityksiä ja tuloksista on ollut hyötyä jatkotutkimuksille.

Avoimien lähdekoodien ohjelmien käyttö on yleistynyt koulumaailmassa. Siitä kerromme enemmän jäljempänä luvussa 8.3.

Kirjastomaailmassa halutaan siirtyä avoimiin ohjelmistoihin sen tähden kun suljettujen ohjelmistojen valmistajat ovat haluttomia tekemään nopeita korjaus-päivityksiä tai uusia haluttuja ominaisuuksia tietokantaohjelmiin[37]. Ne ovat kirjastoille kiinteästi ydintoimintaa.

Juhta

Vuonna 1988 Suomessa perustettiin julkisen hallinnon tietohallinnon neuvottelukunta JUHTA[38]. Sen toiminta perustuu asetukseen, jota on muutettu vuosina 1994, 2003 ja 2006. JUHTA:n tarkoitus on koordinoida, raportoida ja edistää valtionhallinnon ja kuntien hallinnon tietoyhteiskuntakehitystä Suomessa. Lisäksi JUHTA antaa JHS-suosituksia[39], antaa malliesimerkkejä hyvistä käytänteistä (good practices) ja käynnistää pilottiprojekteja. Suosituksia on tehty syksyyn 2007 mennessä noin 40 kappaletta. Se on Kuntaliiton pysyvä yhteistyö- ja neuvotteluelin. Edistämisen lisäksi tarkoituksena on varmistaa palvelujen saanti kansalaisille koko maassa ja vielä mahdollisimman tehokkaasti. Kunnat ottivat käyttöön 8.10.2007 alkaen JIT-2007 sopimusehdot[40]. Tämä yhdenmukaistaa ja uudistaa kunnallisen alan it-hankintojen sopimusehdot. Tämä yhtenäistää julkisen sektorin ja kuntien sopimusehdot ja vähentää hankintaehtojen tulkintatarvetta.

JUHTA suosittelee hallinnolliselle sektorille avoimia ohjelmia kustannuksiltaan tehokkaina ja vapaina lisensseistä. Avointen ohjelmien käyttö vaatii kehittynyttä liiketoimintamallia, koulutusta oppaita ja työkaluja. Lisääntyvä yhteistyö eri hallinnonalojen kesken (jopa globaalisti), riippumattomuus yhdestä toimittajasta ja tulevaisuuden visiot tarvitsevat avointen ohjelmien lisäksi avoimia standardeja.

37 http://pro.tsv.fi/stks/signumnew/200504/2.pdf Avoimeen arkkitehtuuriin (4.10.2006)

38 http://www.intermin.fi/juhta JUHTA:n kotisivu (4.10.2007)

39 http://www.jhs-suositukset.fi/ (4.10.2007)

40 http://www.digitoday.fi/data/2007/10/08/Kunnat+mukaan+julkisen+sektorin+it-hankintoihin/200724834/66 (8.10.2007)

8.2 Suomen viranomaisten kyky käyttää ODF:ää

Huolimatta avoimien dokumenttien standardoinnista Suomen viranomaiset ovat kykenemättömiä käsittelemään sitä muotoa TietoViikko-lehden tekemän kyselyn mukaan. Lokakuussa vuonna 2007 Suomessa vain yksi ministeriö kykeni avaamaan ODF-asiakirjan. (Junkkaala, 2007) Onneksi tilanne on sellainen, että PDF tai XML-muodossa he pystyvät lukemaan lähetetyn asiakirjan.

Kykenemättömyys lukea ODF-asiakirjoja on todennäköisesti vain välivaihe JUHTA:n ja JSF:n ansiosta. Nyt ollaan vielä tilanteessa, että ODF-asiakirjoja ei ole aiemmin käytetty hallinnollisissa tehtävissä. Se on ollut vain vaihtoehto. Aiempien vuosien avoimien koodien ohjelmien tutkimukset keskittyvät vaihtoehdon esittämiseen suljetuille koodeille ja toiseksi GNU-lisenssin ohjelmien ilmaisuus ohjelmistojen käytön kustannuksia vähentävänä. Nyt on siirrytty seuraavaan vaiheeseen: avoimien standardien tuottamien tietojen tulee toimia käytännössä (kuten suljettujenkin tähän asti).

Siirtyminen ODF-dokumentteihin on vielä iso mullistus hallinnollisessa tietojenkäsittelyssä. Standardin vaihto tuonee esiin yhteensopimattomia. Ihmisten muutosvastarinta uusille toimintatavoille voi tulla koetukselle. Aiemmin ”vapauttavaksi” koettu avoimuus näkyykin muutosvaiheessa lisätyötä vaativana vaivana.

8.3 Koulumaailma

Tarkastelemme tässä luvussa avoimia teknologioita suomalaisessa koulumaailmassa. Koulumaailman siirtyminen avoimiin ja ilmaisiin ohjelmiin on alkanut noin vuonna 2004. Tässä siirtymisellä tarkoitetaan ensisijaisesti luokkakäytössä olevia tietokoneita, sillä niiden lukumäärä on suurin kouluissa. Palvelinkoneissa esimerkiksi www-palvelimen käyttöjärjestelmänä on perinteisesti ollutkin Linux. Ilmaiset ohjelmistot ovat ratkaisu niukkeneviin taloudellisiin resursseihin[41]. Lisenssimaksut ovat pienissä kyläkouluissakin niin suuria, että sillä rahamäärällä voisi palkata jopa useita opettajan virkoja. Vastaavasti laitteistoja voitaisi uusia kokonaisia luokallisia. Avoimet ohjelmat leviävät koulumaailmaan lähinnä taloudellisten kustannuksien säästöjen saamiseksi.

Tietoturvahaavoittuvuudet ovat merkittävässä asemassa koulumaailmassa. Riskejä haavoittuvuuksiin tuovat oppilaiden itse asentamat ohjelmat, jos sellainen on sallittu.

41 http://www.opettaja.fi/pls/portal/docs/PAGE/OPETTAJALEHTI_EPAPER_PG/2006_41/101960.htm Noormarkun tietotekniikan uusi ajanlasku (13.10.2006)

Lisäksi esimerkiksi pikaviestimien kautta kulkeutuu paljon viruksia ja muita haittaohjelmia. Yleisesti käytetyssä Microsoft Windows -järjestelmän Internet Explorer -selaimen kautta järjestelmään voi pesiytyä haittaohjelmia ilman, että käyttäjä tekisi mitään ylimääräisiä toimenpiteitä kuin selailisi tavanomaisia www-sivuja.

Avoimien ohjelmien käyttöjärjestelmätasolla kouluhallintoon estää sellainen seikka, että monet kouluhallinnon ohjelmistot kuten lukujärjestysohjelmat, opiskelijoiden suoritusrekisteriohjelmistot jne. toimivat Windows-käyttöjärjestelmällä. Muutoksia jarruttaa lisäksi suurten organisaatioiden jäykkyys ja hitaus muutoksiin. Olemassaolevaa toimivaa ei kannatakaan vaihtaa uuteen, jonka toimivuudesta ei ole takeita. Siinä pienet käyttöongelmat eivät kuitenkaan saa aikaan suuren luokan muutosta.

Suomen Linux-yhdistyksen Flugin sivustolla on Linux kouluissa -sivu osoitteessa

http://www.flug.fi/linuxkouluissa/.

Raportteja ja aktiviteettia projektissa on ollut viimeiksi www-sivun mukaan vuonna 2005 eli tämän tutkimuksen kirjoitushetkellä 3 vuotta sitten.

COSS seuraa myös avoimien ohjelmien leviämistä ja käyttöä koulumaailmassa. Yksi menestystarina on Vitikkalan koulu Jämsässä. Yksi säästö tässä esimerkissä syntyy kun vanhoja koneita ei tarvitse uusia nopeammiksi, vaan niitä voidaan käyttää jäljempänä esiteltävällä LTSP-järjestelmällä. Case on esitelty COSS:in www-sivustolla osoitteessa

http://www.coss.fi/web/coss/casesnnewsletters/1-06linux.

Linux Journal -julkaisssa vuodelta 2004 on mielenkiintoista luettavaa Manitoban suurimman high school -koulun siirtymisestä avoimiin ohjelmiin Kanadassa. Artikkeli on kahdessa osassa osoitteissa

http://www.linuxjournal.com/article/7418

http://www.linuxjournal.com/article/7419.

Opettajille suunnattu avoimien ohjelmistojen ohjeita, ideoita ja ajatuksia on Linux kouluun -sivusto. Materiaali on maksuttomasti kaikkien kopioitavissa. Materiaalit ovat Creative Commons -lisensoituja. Sivusto löytyy osoitteessa

http://www.linuxkouluun.fi/.

LTSP

1990-luvun loppupuolella Jim McQuillan halusi liittää 35 uutta pääteasemaa AS/400 ja SCO Unix palvelinta. He eivät halunneet käyttää tähän Windowsia. He halusivat käyttää tähän Linux-järjestelmää. LTSP-järjestelmässä pääosa tietojenkäsittelystä tapahtuu tehokkaassa keskuskoneessa. Kullakin käyttäjällä on pc-työasemalla vain vähän resursseja kuluttava pääteohjelma, joka hoitaa lähinnä graafisen käyttöliittymän rutiinit ja käyttäjän syöttämät tiedot X-ikkunointijärjestelmän kautta. Tarkoituksena oli luoda halpa järjestelmä usealle tietokoneen käyttäjälle. (LTSP, 1999)

LTSP-järjestelmää varten on kehitetty erityisiä halpoja tähän käyttöön tarkoitettuja tietokoneita. LTSP-järjestelmällä voidaan luokkahuoneeseen järjestää yksi palvelinkone, jota luokallinen oppilaita käyttää.

LTSP-järjestelmän käytön menestystarinoita on luettavissa osoitteessa

http://wiki.ltsp.org/twiki/bin/view/Ltsp/SuccessStories.

9 Yhteenveto

Yhteiskunta muuttui agraariyhteiskunnasta teollisuusyhteiskunnaksi 1800-1900 -luvuilla. Vuosisatojen ja vuosituhannen taitekohdassa yhteiskunta on muuttumassa teollisuusyhteiskunnasta tietoyhteiskunnaksi. Tietojen ja niiden käsittelyn määrä on kasvanut. Tietojen käsittelyn automatisoimisella on ollut monenlaisia vaikutuksia yhteiskunnan kehittymiseen.

Olemme tarkastelleet tutkimuksessa avoimia teknologioita avoimen lähdekoodin, avoimien dokumenttien näkökulmasta. Avoimet ohjelmistot ja dokumentit poistavat esteitä tietojen automaattisessa käsittelyssä. Avoimet teknologiat mahdollistavat hyvien ideoiden, innovaatioiden leviämistä. Näin avoimet teknologiat ovat yhteiskunnan askel eteenpäin. Teknologiat mahdollistavat tiedon vaivattoman tuottamisen, käsittelyn ja levittämisen. Tiedon käsittely abstrahoituu ja tiedon jalostusaste kasvaa määrän ohella. Kehitystä voi verrata kirjapainotaidon kehittymiseen. Voimme kuvitella, mitä avoimet teknologiat Internetin tiedonvälityksineen mahdollistaa tulevaisuuden tietoyhteiskunnassa.

Avoimien teknologioissa avoimen lähdekoodin hyviä puolia ovat laatu, vakaus, turvallisuus ja tehokkuus. Avointen dokumenttien hyviä puolia on riippumattomuus käytetystä ohjelmasta. Muunnoksiin käytetyt resurssit vapautuvat parempaan käyttöön. Ilmaiset ohjelmat mahdollistavat osaltaan maapallon köyhien alueiden kehittymistä. Luonnossa monimuotoisuudella on tarkoituksensa. On syytä tehdä tutkimuksia, voivatko avoimet teknologiat tuoda mukanaan haittoja tai jopa vaaroja tulevaisuudessa. Tällaisia vaaroja voivat olla esim. tietokonevirukset, joiden leviäminen helpottuu yhtenäisissä ympäristöissä.

Avointen ohjelmien kehittäminen on synnyttänyt avoimen kehitysympäristön. Se on laajempi ja löyhempi kuin suljettujen koodien kehitysympäristöt. Näin ollen voidaan ajatella, että se olisi ohjelmistoteknisin mittarein laadultaan parempi suljettujen ohjelmistojen kehitysympäristöihin verrattuna.

Avoimuuteen liittyy usein ilmaisuus. Näin köyhille ihmisille ja maanosille syntyy mahdollisuus. Tekniikan kasvu, sosiaalisuuden kasvu, hyvinvoinnin kasvu. On jännittävää nähdä, mitä hyppäys uuteen teknologiaan saa aikaan. Ehkä he omaksuvat sen luonnollisesti vähin vahingollisin sivuvaikutuksin. Laajempana kysymyksenä voi pohtia, paraneeko ihmiselämä laadultaan.

Tallennusmedioiden muodot näyttävät muuttuvan kiihtyvällä tahdilla. Muutossuunta on yhteensopivuuden kasvaminen ja lisäominaisuuksien käyttöönotto. On entistä tärkeämpää, että avoimet tallennusmuodot tulevat yleiseen käyttöön. Niiden tulee olla

sellaisia, että tietoa ei katoa tallennuksessa eikä muunnoksissa toisiin muotoihin. Avointen dokumenttien standardeita on kolme erilaista. Tähän ovat johtaneet historialliset organisaatioista johtuvat seikat ja myös kaupallisella puolella omia etujaan ajavat yksittäiset yritykset. Avoin standardi ei takaakaan täydellistä yhteensopivuutta järjestelmien kesken. Tarvitaan yhä uutta standardoimiskehitystä.

Avoimet teknologiat kehittävät ohjelmistoja ja dokumentteja helpommin käytettäviksi. Kehitys jatkuu varmasti jossain. Sitä kehitystä avoimet teknologiat tukevat ja mahdollistavat omalta osaltaan.

Viitteet

Aalto-Setälä Sauli & Leppäniemi Jari (2005) *F/OSS – vapaan ja avoimen lähdekoodin ohjelmistot, esiselvitys*, Tampereen teknillinen yliopisto, Porin yksikkö julkaisu 4 / 2005.

BUGZILLA (1998) *Bugzilla home*, http://www.bugzilla.org/ (1.6.2008)

COSS (2003) *Suomen open source -keskus*, http://www.coss.fi/ (1.6.2008)

CVS (2000) *CVS:n kotisivu* http://www.cvshome.org/ (20.9.2003).

Fanderl H., Fischer K., Kämper J. (1992) The Open Document Architecture: From standarization to the market, *IBM Systems Journal*, **31**(4), 728-754.

FSF (1985) *Free Software Foundation* http://www.fsf.org/ (1.6.2008)

GIMP (2008) *The GNU Image Manipulation Program* http://www.gimp.org/ (30.5.2008)

GNU (1984) GNU:n kotisivu http://www.gnu.org/ (30.5.2008), suomenkielinen kotisivu http://www.gnu.org/home.fi.html (30.5.2008)

Haikala, I., Märijärvi, J. (1997) *Ohjelmistotuotanto*. Suomen Atk-kustannus, Gummerus, Espoo. 4. uusittu painos.

Hoek, van der Hoek, A., Hall, R. H., Heimbigner, D., Wolf, A. L. (1997) Software Release Management, *Software Engineering Notes,* **22**(6), 161-175.

Joukahainen (2006) *Joukahainen* http://joukahainen.lokalisointi.org/ (23.1.2007).

Junkkaala Jouni (2007) Avoin ei aukea hallinnossa, *TietoViikko*, **25**(33), 2-3.

Korpela Jukka (2005) *Siirtokirjoitus (translitterointi ja transkriptio) ja sen suomalaiset käytännöt* http://www.cs.tut.fi/~jkorpela/kielet/siirtokirjoitus.html (11.6.2008).

KDE (1998) *K-desktop environment* http://www.kde.org/ (1.6.2008)

Koponen Timo (2007) *Evaluation of Maintenance Process in Open Source Software Projects Through Defect and Version Management Systems*, Kuopion yliopisto. ISBN 978-951-781-988-6, 978-951-27-0107-0 (PDF), ISSN 1459-7586.

Kotoistus (2004) *Kotoistushanke* http://www.kotoistus.fi/ (2.6.2008)

Larsson Peter (2008) *Hyllar Finland -sågar Linus*. ComputerSweden http://www.idg.se/2.1085/1.147578 (30.5.2008)

LTSP (1999) Linux Terminal Server Project http://www.ltsp.org/ (15.6.2008)

Microsoft (2008) *Microsoft Expands List of Formats Supported in Microsoft Office* http://www.microsoft.com/Presspass/press/2008/may08/05-21ExpandedFormatsPR.mspx (10.6.2008)

NASA NAS Technical Reports (1989) *Developing an Open Source Option for NASA Software* http://www.nas.nasa.gov/News/Techreports/2003/PDF/nas-03-009.pdf (8.3.2007).

ODF (2005) *Open Document Format* http://en.wikipedia.org/wiki/OpenDocument (27.1.2007).

Ollila Kauko (2008) *Uutiskommentti Tietoviikko-lehdessä 26.3.2008* http://www.tietoviikko.fi/doc.te?f_id=1331589 (10.6.2006)

OOXML (2008) *Office Open XML* http://en.wikipedia.org/wiki/Office_Open_XML (11.6.2008)

OpenOfficeOrg (1999) *OpenOffice.org:in kotisivu*. http://www.openoffice.org/ (23.1.2007).

OSI (1998) *Open Source Initiative* http://www.opensource.org/ (31.5.2008)

OSSCENSUS (2008) *The Open Source Census* http://www.osscensus.org/ (31.5.2008)

OSOR (2005) *OSO-R Project* http://osor.eu/ (15.6.2008)

OSTG *Open Source Technology Group* http://www.ostg.com/about/ ohjautuu edelleen osoitteeseen http://web.sourceforge.net/ (2.6.2008).

Ousterhout, K. (1994) *TCL and the TK Toolkit*. Addison-Wesley Publishing Company, Reading (MA) Yhdysvallat.

Riikonen Matti (2006) *Ohjelmistojen lokalisointi ja kansainvälistäminen*, Pro Gradu -tutkielma, Joensuun yliopisto (30.5.2008)

Sektori (2003) EU suosittelee avoimen lähdekoodin järjestelmiä

http://sektori.com/uutinen/eu-suosittelee-avoimen-lähdekoodin/4763/ (15.6.2008)

SourceForge (2004) SourceForgen kotisivu http://www.sourceforge.net/ (13.6.2008)

Spinellis & Szyperski (2004) How Is Open Source Affecting Software Development? *IEEE Software* **04**(2004), 28 – 33.

SVN (2000) *Subversionin kotisivu*, http://subversion.tigris.org/ (30.5.2008)

Teräs Arto & Tolonen Pekka (2000) Nörttien Linux myyntipakettiin. *MikroPC* **2000**(5), 44 – 50.

TIGRIS (1997) *Tigris.org Open Source Software Engineering Tools* http://www.tigris.org/ (1.6.2008)

Voikko (2006) *Voikko – suomen kielen tavutus- ja oikeinkirjoitusohjelmisto* http://www.lemi.fi/voikko/ (23.1.2007).

Wikipedia (2008) *Ohjelmistotuotanto* http://fi.wikipedia.org/wiki/Ohjelmistotuotanto/ (1.6.2008)