TURUN YLIOPISTO
Kauppakorkeakoulu

Ohjelmisto-omaisuuden hallinta suurissa yrityksissä

Tietojärjestelmätieteen kandidaatintutkielma

Laatija:
Joona Melametsä

Ohjaaja:
KTT Jonna Järveläinen

8.12.2023
Turku

Kandidaatintutkielma

Oppiaine: Tietojärjestelmätiede
Tekijä: Joona Melametsä
Otsikko: Ohjelmisto-omaisuuden hallinta suurissa yrityksissä
Ohjaaja: KTT Jonna Järveläinen
Sivumäärä: 25 sivua
Päivämäärä: 8.12.2023

Ohjelmisto-omaisuuden hallinta on prosessi, jonka avulla yritykset hallinnoivat ja optimoivat ohjelmistojensa käyttöä. Se sisältää ohjelmistoinventaarion ylläpidon ja ohjelmistolisensoinnin seurannan. Lisäksi sen avulla varmistetaan, että kaikki käytössä olevat ohjelmistot ovat asianmukaisesti lisensoituja. Sen hallinta auttaa yrityksiä välttämään oikeudellisia riskejä ja hallitsemaan IT-kustannuksia tehokkaammin. Ohjelmisto-omaisuuden hallinta on keskeistä yritysten strategisessa päätöksenteossa ja resurssien optimoinnissa. Se vaatii jatkuvaa seurantaa ja päivitystä vastaamaan yritysten muuttuvia tarpeita.

Ohjelmisto-omaisuuden hallinnan tehokkuus yrityksissä perustuu sen kykyyn mukautua jatkuvasti muuttuvaan teknologiseen ympäristöön ja liiketoiminnan vaatimuksiin. Tämä prosessi kattaa ohjelmistoinventaarion ja lisensoinnin hallinnan lisäksi, myös ohjelmistojen elinkaaren hallinnan aina hankinnasta poistoon. Strateginen lähestymistapa tähän hallintaan mahdollistaa yrityksille paremman kontrollin IT-budjettiinsa. Näin varmistetaan, että ohjelmistoinvestoinnit tuottavat parhaan mahdollisen hyödyn. Keskeistä on myös ohjelmistojen käyttöasteen analysointi, joka auttaa tunnistamaan tarpeettomia ohjelmistoja ja eliminoimaan niistä aiheutuvat kustannukset. Yritykset hyödyntävät ohjelmisto-omaisuuden hallintaa myös riskienhallinnassa. Riskienhallinnalla vältetään lisensointiin ja yhteensopivuuteen liittyviä ongelmia, jotka voivat aiheuttaa sekä taloudellista että oikeudellista haittaa. Tämän järjestelmällisen lähestymistavan avulla yritykset voivat lisätä IT-toimintansa läpinäkyvyyttä ja tehostaa päätöksentekoa ohjelmistoresurssien osalta.

Ohjelmisto-omaisuuden hallintatyökalut auttavat ylläpitämään tarkkaa tietokantaa kaikista organisaation käyttämistä ohjelmistoista. Tunnistamalla käyttämättömät tai alikäytetyt ohjelmistot voidaan saavuttaa merkittäviä kustannussäästöjä. Lisäksi ne voivat tarjota ominaisuuksia, kuten automaattisen lisenssien seurannan ja vaatimustenmukaisuuden hallinnan. Näin voidaan varmistaa, että kaikki ohjelmistot ovat asianmukaisesti lisensoituja. Työkalut voivat myös tukea yritysten strategista päätöksentekoa, resurssien optimointia, tehostaa IT-budjetin hallintaa ja lisätä IT-toimintojen läpinäkyvyyttä.

Avainsanat: ohjelmisto-omaisuus, ohjelmisto-omaisuuden hallinta, ohjelmisto-omaisuuden hallintatyökalut

SISÄLLYS

KUVIOT

TAULUKOT

1 Johdanto

Ohjelmistoilla on nykyaikaisessa yritysmaailmassa merkittävä rooli lähes kaikissa organisaatioissa. Ne ovat välttämättömiä työkaluja monille työntekijöille ja tästä syystä optimaalisen ohjelmiston käytettävyyden merkitys korostuu. Ohjelmistojen ja ohjelmistopohjaisten palveluiden kuten SaaS (software as a service) ja PaaS (platform as a service) hankinnat muodostavat yleisesti huomattavan osuuden tietotekniikkabudjetista. Kyselytutkimus, joka kohdistettiin 501 IT-johtajalle ympäri maailmaa, paljasti, että jopa yli 50 prosenttia IT-budjetista ohjataan ohjelmistojen ja niihin liittyvien resurssien hankintaan. (Flexera, 2023.)

Tehottomat ohjelmistonhankinta- ja hallintakäytännöt voivat seurauksenaan aiheuttaa monenlaisia haasteita, kuten kustannusten kasvua, lisenssisopimusten rikkomuksia, puutteellisen käsityksen ohjelmistojen määrästä sekä altistaa kyberturvallisuusriskeille. Ilman tarkkaa käsitystä ohjelmistolisenssien määrästä ja niiden käyttäjämäärästä kunkin ohjelmiston osalta, voi johtaa organisaation jakamaan resurssejaan tehottomasti. Tämä puolestaan saattaa johtaa ohjelmistolisenssien alikäyttöön tai ylikäyttöön. Riittämätön lisensointi voi myös altistaa organisaation vaatimustenmukaisuusriskeille (engl. compliance). (Varela, Méxas & Drumond, 2018.)

Tutkielman tutkimuskysymykset ovat:

1. Millaiset prosessit ja käytännöt ovat hyödyllisiä ohjelmisto-omaisuuden hallinnan optimoinnissa?

2. Mitä vaikutuksia ohjelmisto-omaisuuden hallintatyökaluilla voi olla suurille yrityksille?

Tutkielmassa keskitytään erilaisiin prosesseihin ja käytäntöihin, jotka ovat tarpeellisia suurissa yrityksissä ohjelmisto-omaisuuden hallinnan tehostamisessa. Tutkielmassa tullaan myös tarkastelemaan ohjelmisto-omaisuuden hallintatyökaluja ja niiden vaikutuksia suurissa yrityksissä. Työkalujen tarkastelussa keskitytään lisenssien seurantaan, käyttöoikeuksien hallintaan ja turvallisuusnäkökulmaan.

2 Ohjelmisto-omaisuuden hallinta

Ohjelmisto-omaisuuden hallinta (Software Asset Management, SAM) on yrityksen tai organisaation prosessi, jolla hallitaan ja optimoidaan hankittujen ohjelmistotuotteiden ostoa, käyttöä, ylläpitoa ja hävittämistä. Se sisältää ohjelmistoinventaarion hallinnan ohjelmistolisenssien hallinnan, varmistaa lisensointiehtojen noudattamisen ja pyrkii minimoimaan riskejä. (Holsing & Yen, 1999.) Ohjelmistojen käyttömäärän näkyvyyden puute on keskeinen tekijä ohjelmistojen kustannusten nousussa maailmanlaajuisesti (McCarthy & Herger, 2011). Sen avulla organisaatiot voivat välttää ylimääräisiä kustannuksia, parantaa IT-infrastruktuurin tehokkuutta ja varmistaa ohjelmistojen lisensoinnin oikeellisuuden (Varela, Méxas & Drumond, 2018).

2.1 Ohjelmisto-omaisuuden hallinnan prosesseja

McCarthy ja Herger (2011) ovat esittäneet ratkaisun yhdistämään IT:n, prosessit ja liiketoiminnan ohjelmisto-omaisuuden hallinnan näkökulmin. Se sisältää neljä kohtaa: Ohjelmisto-omaisuuden löytämisen, joka koostuu asennettujen lisenssien skannauksesta; hankittujen omaisuuserien yhteensovittamisen, joka helpottaa hankittujen kolmansien osapuolien ohjelmistolisenssien elinkaaren hallintaa; sopimustenhallinnan toteuttamisen sekä liiketoimintatiedon raportoinnin, jonka avulla seurataan IT-ohjelmistokulutusta ja tuotetaan tietoa auditointivalmiutta varten.

ISO/IEC 19770-1 on kansainvälisesti hyväksyttyjen International Organization for Standardization (ISO) ja International Electrotechnical Comissionin (IEC) luoma standardi, joka koskee ohjelmisto-omaisuuden hallintaa (Albert, Dos Santos & Werner, 2013). ISO/IEC 19770-1 (2012) jakaa ohjelmisto-omaisuuden hallinnan neljään tasoon: Luotettava tieto, käytännön hallinta, toiminnallinen integraatio ja täysi ISO/IEC yhteensopivuus ohjelmisto-omaisuuden hallintaan.

Luotettavan tiedon tason tavoitteena on saavuttaa relevanttia tietoa siitä, mitä ohjelmisto-omaisuutta organisaatiolla on hallussaan. Tämä taso kattaa myös ohjelmistolisensointiin liittyvät toiminnot kuten lisenssien noudattamisen.

Käytännön hallinnan tavoitteet perustuvat ensimmäisen tason tietoihin. Siinä organisaatio on tunnistanut ongelmat tietojen puutteessa tai laadussa sekä parannus- ja

säästömahdollisuudet. Tämän jälkeen organisaatio pyrkii parantamaan hallinnan valvontaa sekä saavuttamaan välittömiä hyötyjä.

Toiminnallisen integraation taso keskittyy operatiivisen integraation parantamiseen kehittäen ohjelmisto-omaisuuden hallinnan tehokkuutta organisaatiossa. Se perustuu kahteen edelliseen tasoon ja sisältää integraatioprosesseja sopimustenhallintaan. Se kattaa myös ohjelmiston hankinta- käyttö- ja poistovaiheet.

Taso neljä keskittyy saavuttamaan ihanteellisen tilanteen, jossa ohjelmisto-omaisuuden hallinta on integroitu organisaation strategiseen suunnitteluun. Strategisen vaiheen saavutettuaan organisaation ohjelmisto-omaisuuden hallintaprosessi tukee strategisia liiketoimintatavoitteita. Tässä tilanteessa organisaatiolla on mahdollista saavuttaa kustannusten vähennyksiä, tuotannon kasvua ja kilpailukykyisiä innovaatioita. Taso neljä on optimaalinen vaihe ohjelmisto-omaisuuden hallintaprosessissa.

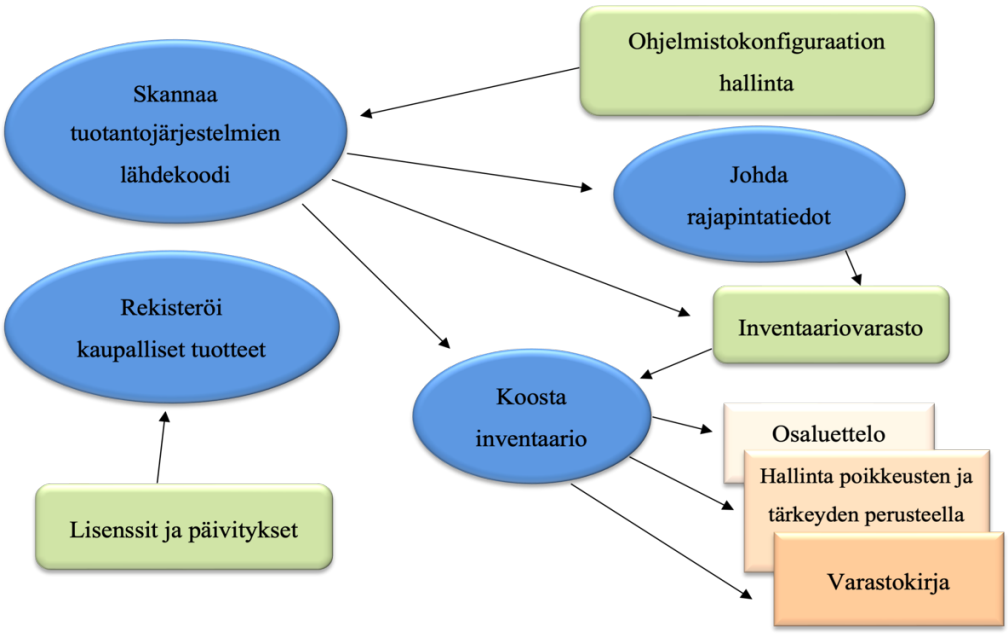
Seuraavissa alaluvuissa tullaan keskittymään kahteen edellä mainittuun prosessiin: inventaarion hallintaan ja lisenssien hallintaan. Ne kuuluvat ISO/IEC 19770-1 standardin ensimmäiseen tasoon sekä liittyvät vahvasti McCarthy ja Hergerin (2011) neljän kohdan ratkaisuun. Inventaarion hallinta ja lisenssien hallinta ovat olennaisia prosesseja luomaan perustan toimivalle ja laadukkaalle ohjelmisto-omaisuuden hallinnalle (Klint & Verhoef, 2002; Ben-Menachem & Marliss, 2004; McCarthy & Herger, 2011).

2.2 Inventaarion hallinta

McCarthy ja Herger (2011) ovat listanneet tietämättömyyden käytössä olevista ohjelmistoista yhdeksi tärkeäksi ongelmaksi suurissa organisaatiossa. Jotta organisaatio on tietoinen sen ohjelmisto-omaisuudesta, tulee sen koostaa inventaario käytössä olevista ohjelmistoista. Puutteet inventaariotiedossa rajoittaa organisaation tietoisuutta IT-menoihin liittyen. Tiedonhallinnalla on mahdollista ratkaista kulutukseen liittyviä ongelmia, ja vielä merkittävämpää vähentää turhia menoja. (Klint & Verhoef, 2002.)

Ben-Menachem ja Marliss (2004) ovat esittäneet muutoksen paradigman (engl. the paradigm of change), joka perustuu työkaluihin, menetelmiin ja menettelytapoihin sopivan kokonaisvaltaisen IT-inventaarionhallinnan kannalta. Kirjoittajien mukaan ohjelmistoinventaario on muutakin kuin lisenssien hallintaan tarkoitettu ohjelmistojen luettelo. Se on tarkasti jäsennelty yksityiskohtainen luettelo arvokkaista omaisuuseristä. Lisäksi se on tietovarasto, joka tarjoaa tukea ohjelmistojen hallintaan ja kehitykseen.

Inventaarion rakentaminen aloitetaan keräämällä tietoja suoraan ohjelmien lähdekoodista. Prosessi kategorisoi ohjelmistokohteiden tiedot järjestelmittäin ja alijärjestelmittäin. Tämän jälkeen se kartuttaa näitä tietoja toiminnallisilla kuvauksilla ja järjestelmävaatimuksilla. Kuva 1 esittää tarvittavat toimenpiteet ohjelmistoinventaarion luomiseen.



Kuva 1. Inventaarion luonti ohjelmisto-omaisuudelle (Ben-Menachem & Marliss, 2004)

Ben-Menachem & Marliss (2004) mukaan IT:n merkittävimmät epäonnistumiset ovat järjestelmien puute, jotka tukevat, keräävät ja tarjoavat tietoa ohjelmistokohteiden hallintaan. He esittävät yhdeksi yleiseksi virheeksi IT-ammattilaisilla heidän taipumuksensa ajatella ohjelmistonhallintaa versio- tai konfiguraationhallinnan termein. Versionhallintajärjestelmillä ja ohjelmistokonfiguraationhallintajärjestelmillä on kuitenkin tarkoitus hallita yksittäisten ohjelmisto-objektien versioita. Kyseinen ajattelutapa ei ole ratkaisu suurten ohjelmistomäärien hallinnan ongelmien ratkaisemiseen. Lisäksi kirjoittajat esittävät, että ohjelmistoinventaario on osa laajempaa teknologian kokonaisuutta, jotka on kehitetty ratkaisemaan teknologian ja liiketoimintaprosessien jatkuvia muutoksia. Tämän takia sijoitus ohjelmistoinventaarion luomiseen ja ylläpitoon on ratkaisevan tärkeää asianmukaiselle pitkäaikaiselle ohjelmisto-omaisuuden hallinnalle.

Ben-Menachem & Marliss (2005) on määrittänyt menetelmän ohjelmistojen hallintaan jo valmiiksi teollisuustekniikassa vakiintuneen metodologian hallinta tärkeyden ja poikkeuksen perusteella (engl. control by importance and exception). Prosessi sisältää kohteiden luokittelun, jokaisen kohteen arvon laskemisen ja arvokertoimen määrittämisen. Kohteiden luokitteluprosessissa ohjelmistot jaetaan ryhmiin. Ryhmillä on ominaisuuksia, jotka auttavat niiden ymmärtämisessä ja käytössä. Ryhmät voidaan jakaa attribuuttien hierarkkisen riippuvuuden, ohjelmiston ominaisuuksien, relaatioiden sekä optimaalisen kohdekoon mukaan. Toisessa vaiheessa ohjelmistot luokitellaan arvon mukaan. Ei ole olemassa vain yhtä absoluuttista vaihtoehtoa arvon määrittelemiseen vaan optimaalisen määritelmän kehittäminen on yrityskohtaista. Kirjoittajat suosittelevat tässä yhteydessä arvon määrittelyä ajallisesti, sillä ohjelmistojärjestelmät ovat muuttuvia ja kehittyviä järjestelmiä. Prosessi aloitetaan laskemalla todelliset kustannukset rahallisesti tai ajallisesti. Sen jälkeen arvioidaan hankinta- ja ylläpitokustannukset. Tämän jälkeen lasketaan arvioitu korvauskustannus ohjelmistolle, jonka jälkeen lasketaan vuotuinen käyttö. Lopuksi lasketaan yhteen koko vuoden kustannukset. Arvokertoimen määrittämiseen kuuluu arvon yksi, kaksi tai kolme antaminen edellisen vaiheen laskettujen arvojen perusteella sekä kohteiden luokittelu kategorioihin A, B ja C. A-kategorian kohteet kattavat vuotuisista kustannuksista korkeimman 20- prosenttia, C-kategoria alimmat 20- prosenttia ja B- kategoria keskimmäiset 60- prosenttia. Tämä määrittely määrää luokittelussa kohteiden lopullisen järjestyksen. Tämän metodologian avulla on mahdollista tehostaa ohjelmisto-omaisuuden hallintaa ohjelmistoinventaariota hyväksi käyttäen.

Klint & Verhoefin (2002) mukaan ohjelmistosovellusten inventaario yhdistettynä yksityiskohtaiseen tietoon jokaisesta sovelluksesta mahdollistaa tiedon saamisen koko ohjelmisto-omaisuuden suorituskyvystä, laadusta ja kustannuksista. He korostavat vahvasti sekä tiedolla johtamisen tärkeyttä että sen jakamiseen liittyviä etuja. Taulukko 1 sisältää Klint & Verhoefin (2002) sekä Ben-Menachem & Marlissin (2004) määritelmät optimaalisesta ohjelmistoinventaarion sisällöstä.

Taulukko 1. Ohjelmistoinventaarion sisältö

(Klint & Verhoef, 2002)	(Ben-Menachem & Marliss, 2004)
Yksityiskohtainen luettelo ohjelmistoista	Omaisuuden nimi
Kehityskustannukset	Ohjelmiston tyyppi
Palvelun laatu toiminnan aikana	Toiminnallisuuden perustaso
Toiminnan aikaiset kustannukset	Omaisuuserän sijainti
Ylläpitokustannukset	Luettelot järjestelmistä, joissa omaisuuserää tällä hetkellä käytetään sekä järjestelmätyypeistä, joissa se voisi tulevaisuudessa toimia, mukaan lukien sen projektin tai sovelluksen omistajat
Taloudellinen arvo liiketoiminnan näkökulmasta	Ohjelmiston ikä ja odotettavissa oleva hyödyllinen käyttöikä
Kaikkien kustannustekijöiden laadullinen ja määrällinen arviointi	Tekniset, toiminnalliset ja laadulliset mittarit – esim. määrä, monimutkaisuus ja toiminnallisuus
	Rakennustyökalut (esimerkiksi kieli, generaattori ja linkittäjä), käyttöalusta ja riippuvuudet
	Vaatimustenmukaisuuden tila olemassa olevien yritysstandardien suhteen

2.3 Ohjelmistolisensointi

Ohjelmistolisensointi on käytäntö, jonka avulla käyttäjät voivat hankkia oikeuden asentaa ja käyttää ohjelmistoja laitteella tai verkossa. Lisenssi voidaan myöntää yksittäiselle käyttäjälle, käyttäjäryhmälle tai organisaatiolle. Käyttäjän tulee ostaa useita lisenssejä, jos hän haluaa käyttää ohjelmistotuotetta laillisesti monilla eri laitteilla (Yuan ym., 2022). Lisenssin edellytyksenä on, että käyttäjät noudattavat ohjelmistotoimittajan asettamia käyttöehtoja. Lisensointiprosessin päämäärä ohjelmistokehittäjän näkökulmasta on varmistaa, että sen taloudelliset intressit säilyvät turvattuina estämällä luvaton käyttö ja levitys. Puolestaan asiakkaan näkökulmasta lisenssi suojelee mahdollisilta oikeudellisilta

seuraamuksilta, jotka voivat aiheutua ohjelmistojen virheellisestä käytöstä tai hallinnasta. (Ferrante, 2006.)

Ohjelmistojen väärinkäyttöä kutsutaan ohjelmisto- tai lisensointipiratismiksi. Sillä tarkoitetaan laitonta tai luvatonta ohjelmistojen kopiointi-, levitys- ja lataustoimintaa (Chavarria & Morrison, 2014). Ohjelmistopiratismia on kahdenlaista, kovaa ja pehmeää. Kovalla ohjelmistopiratismilla tarkoitetaan tarkoituksellista ohjelmiston käyttämistä tai uudelleenjakelua huolimatta sen lisensointisuojauksista. Pehmeässä ohjelmointipiratismissa käyttäjä tai käyttäjät hyödyntävät tuotetta tahattomasti tavalla, joka rikkoo ohjelmiston lisenssisopimusta. (Ferrante, 2006.) Tekijänoikeuksien suojaamiseksi ohjelmistolisenssi on yleisesti sidottu käyttäjän identiteettiin tai IP-osoitteeseen (Yuan ym., 2022).

Ohjelmistolisensoinnin lisäksi organisaatioilla on mahdollisuus kehittää yrityksen sisäinen (engl. in-house) ohjelmisto. Yrityksen sisäinen ohjelmisto kehitetään nimensä mukaisesti itse organisaation sisällä eikä sitä hankita markkinoilta. Tämä on yleinen ratkaisu yrityksille, jotka haluavat mukauttaa ohjelmiston omiin erityistarpeisiinsa. (Postmus, Wijngaard & Wortmann, 2009.) Yrityksen sisäisen ohjelmiston käyttöönotossa työmäärä on alkuun suurempi vaatimusten selvittämisen, testaamisen ja palautteen keräämisen myötä. Sen toimeenpano ja koulutus voivat olla kuitenkin nopeampia. Lisäksi sisäisessä kehityksessä käyttäjillä on suora yhteys kehittäjiin, joka nopeuttaa muutosten tekoa. Sisäisesti kehitetyllä ohjelmistolla voi olla myös etuja organisaation tiedon- ja riskienhallinnassa, kun sillä on enemmän kontrollia ohjelmiston datasta ja ohjelmiston elinkaaresta. (Jackson & Brannon, 2018.)

2.4 Ohjelmistolisensointimalleja

Li ym. (2017) listaavat ohjelmistolisensointimalleiksi vaihtoehdot paikan päällä mallin (engl. on-premises model) ja SaaS-lisensointimallin. Paikan päällä mallista on yleisemmin käytetty nimitystä pysyvä malli (engl. perpetual model) ja SaaS-mallista tilauspohjainen malli (Zhang & Seidmann, 2010; Xin, 2020). Pysyvän mallin periaate on, että asiakas ostaa ohjelmiston ja omistaa lisenssin pysyvästi. Tilauspohjainen malli on käytännössä ohjelmiston vuokrausmalli, jossa asiakkaat vuokraavat ohjelmiston ja maksavat ohjelmistotoimittajalle vuokraa käyttöajalta. (Zhang & Seidmann, 2010; Li ym., 2017; Xin, 2020.)

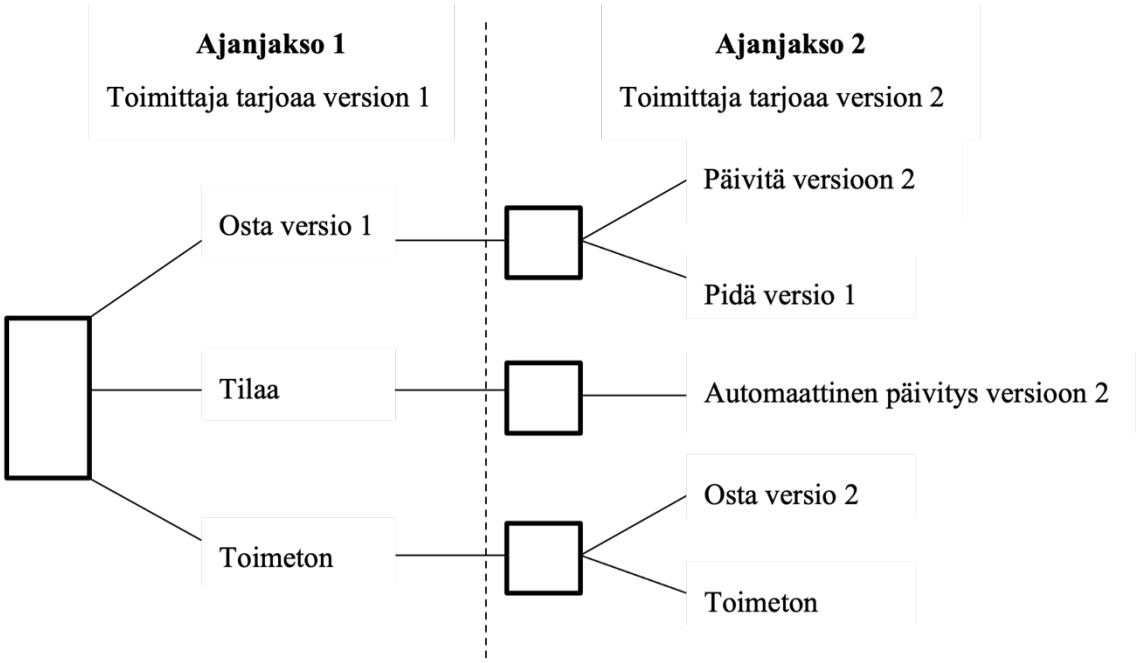
Ferrante (2006) puolestaan lajittelee ohjelmistolisensointimallit seitsemään yleisesti käytettyyn lisenssimalliin, jotka ovat paketoitu, kokeilu, pysyvä, tilauspohjainen, palvelin (prosessoriyksikköä kohden), verkkopohjainen ja käyttöpohjainen malli. Tarkastellaan tarkemmin pysyvää ja tilauspohjaista mallia. Käydään läpi myös muut Ferranten (2006) listaamat mallit, jotta saadaan laajempi kuva mahdollisista lisensointimalleista.

2.4.1 Pysyvä ja tilauspohjainen malli

Pysyvä malli on ollut historiallisesti suosituin lisensointimalli 2000-luvun alkuun saakka (Xin, 2020). Pysyvän mallin omaksuessa asiakas maksaa lisensseistä etukäteen ja saa vastineeksi pysyvät oikeudet ohjelmiston käyttöön. Asiakkaalle myönnetyn pysyvän lisenssin ansiosta hänellä on mahdollisuus käyttää ohjelmistoa rajoittamattomasti. (Zhang & Seidmann, 2010) Kuva 2 esittää kuluttajien vaihtoehdot lisenssien valintaan Zhang ja Seidmannin (2010) mallin mukaan.

Valittaessa pysyvää lisenssimallia tulee ottaa huomioon ylläpidon merkitys. Alkuperäinen maksu kattaa itse ohjelmiston, mutta se ei kata tulevia ohjelmistopäivityksiä, kuten virheenkorjauksia, tietoturvapäivityksiä tai pieniä ominaisuuspäivityksiä. (Ferrante, 2006.) Mikäli ohjelmistotoimittaja julkaisee uuden version tuotteestaan, kuluttajalla on mahdollisuus joko päivittää tuote lisämaksua vastaan tai pitää jo aikaisemmin ostettu versio käytössään (Zhang & Seidmann, 2010).

Tilauspohjaisessa mallissa kuluttaja hankkii lisenssejä määräajaksi, jonka aikana ohjelmiston päivitykset tulevat kuluttajalle automaattisesti ilman lisämaksua (Zhang & Seidmann, 2010). Yleisesti tilauspohjaisen mallin lisenssi hankitaan vähintään vuodeksi. Malli mahdollistaa asiakkaille maksujen jakamisen pidemmälle ajanjaksolle ja ohjelmistotoimittajille vakaan tulovirran (Ferrante, 2006). Tilauspohjainen malli tarjoaa myös parantunutta tehokkuutta toimittajien mittakaavaetujen ansiosta sovelluksen hallinnassa sekä mahdollistaa skaalautuvan käyttökapasiteetin. Tilauspohjaisen mallin käyttöönottaminen on yleistynyt huomattavasti ja se kattoi lähes 50-prosenttia uusien lisenssien myynnistä suurten yritysten sovellusohjelmistomarkkinoilla vuonna 2015. (Xin, 2020.)



Kuva 2. Kuluttajien strategiat (Zhang & Seidmann, 2010)

Tilanteessa, jossa pysyvässä mallissa lisenssit maksetaan etukäteen ennen ohjelmiston saamista, kuluttajalla ei ole mahdollisuutta kokeilla ohjelmistoa ennen sen hankintaa. Tässä tilanteessa ohjelmiston käyttöönottoon liittyy merkittävää arvonluonnin epävarmuutta. Tämän vuoksi monet alalla toimivat asiantuntijat suosivat tilauspohjaista lisensointia pysyvän lisensoinnin sijaan. Tällä tavoin asiakkaiden taloudellinen sitoumus pidetään alhaisempana alkuvaiheessa, kunnes ohjelmiston todellinen arvo on paremmin tunnistettavissa. (Xin, 2020.)

Li ym. (2017) mukaan ohjelmistoihin liittyy myös säädettyjä tapoja harjoittaa liiketoimintaa, jotka eivät välttämättä vastaa asiakkaan olemassa olevia prosesseja. Nämä tulevat esille tilanteessa, kun yritys on ottamassa käyttöön pysyvää lisensointimallia. Yritys joutuu joko uudelleensuunnittelemaan liiketoimintaprosessinsa, jotta se sopii ohjelmiston käytettävyyteen tai tekemään muokkauksia ohjelmiston ominaisuuksissa. Näistä muokkauksista voi syntyä huomattavia lisäkustannuksia. On tärkeää huomioida, että räätälöinnit (engl. customization) eroavat rakenteellisista (engl. configuration) muokkauksista. Räätälöinneillä tarkoitetaan lähdekoodiin liittyviä muokkauksia ja rakenteellisilla muokkauksilla ohjelmiston sisältämiä vaihtoehtoja, esimerkiksi onko päivämäärät Yhdysvaltojen vai Suomen muodossa. SaaS-pohjaisessa tilaus mallissa yrityksellä ei ole mahdollisuutta samanlaisiin räätälöinteihin kuin pysyvässä mallissa. Tämä johtuu siitä, että asiakkaat ovat yhteydessä palvelimeen käyttääkseen ohjelmistoa.

Saas-pohjaisen tilaus mallin muokkaamattomuus on yksi sen merkittävimmistä puutteista yritysten näkökulmasta.

2.4.2 Ohjelmistolisensoinnin eri malleja

Palvelin mallissa lisenssikustannukset määrittyvät ohjelmiston käyttämien CPU:iden eli prosessoriytimien lukumäärästä. Jokainen fyysinen tai virtuaalinen CPU tai ydin lasketaan yhdeksi lisenssiksi. Organisaatioiden on hankittava riittävä määrä lisenssejä kattamaan järjestelmänsä CPU:t tai ytimet. (Ferrante, 2006.) Paketoidussa lisenssimallissa ostaja saa ohjelman käyttövalmiina. Siinä yksi lisenssi hankitaan yhdelle käyttäjälle tai koneelle. Kokeilu malli ja pysyvä malli ovat paketoituja malleja. (Ferrante, 2006.)

Kokeilu malli toimii nimensä mukaisesti. Käyttäjät voivat testata ohjelmistoa maksutta ilman lisenssiä ennen sen ostamista. Kokeiluversioissa ohjelmistoilla voi olla sekä toiminnallisia että aikarajoituksia. Toiminnallinen rajoitus pienentää niiden ominaisuuksien määrää, joita on saatavilla kokeiluversiossa. Aikarajoitus taas määrittelee sen ajanjakson, jonka käyttäjä voi ohjelmistoa kokeiluversiona käyttää. (Li ym., 2017.) Aikarajoitteisen kokeiluversion kesto on yleisesti 30 ja 90 päivän väliltä. Kokeiluversioihin kuuluu harvoin minkäänlaista asiakaspalvelua tai tukea (Ferrante, 2006). Xin (2020) on todennut, että kokeilu version avulla on mahdollista vähentää arvon epävarmuutta, mutta käyttö kontrolloidussa ympäristössä ei altista yritystä operatiivisten olosuhteiden haasteille.

Verkkopohjaisessa mallissa ohjelmisto asennetaan palvelimelle, jossa sen käyttö tapahtuu verkon välityksellä useiden käyttäjien toimesta. Lisenssien hallinta suoritetaan yleensä lisenssipalvelimen avulla, joka valvoo käyttäjien samanaikaista määrää. Tämä varmistaa, että käytössä olevien lisenssien määrä pysyy ostettujen lisenssien sallimissa rajoissa. (Ferrante, 2006.) Käyttöpohjaisessa mallissa yritykset hankkivat tarpeidensa mukaisen määrän lisenssejä. Hinnoittelu perustuu ohjelmiston todelliseen käyttömäärään, joka mitataan joko käyttäjien määrän tai käyttötapahtumien lukumääränä (Postmus, Wijngaard & Wortmann, 2009). Tämä lisenssimalli sopii yhteen samanaikaisesti verkkopohjaisen mallin kanssa. Verkkolisenssien hallintajärjestelmä pystyy seuraamaan jatkuvasti kunkin käyttökerran ja päivittämään sen tilaa, jotta tunnistetaan, milloin käyttäjät ylittävät sallitut rajat. (Ferrante, 2006.)

2.4.3 Lisenssimallien vertailua

Vertaillaan lisenssimalleja keskenään, jotta havaitaan eri mallien hyötyjä ja mahdollisia haasteita. Paketoidussa mallissa yksi lisenssi hankitaan kullekin käyttäjälle tai koneelle ja käyttäjät saavat ohjelman käyttövalmiina (Ferrante, 2006). Tämä tekee siitä yksinkertaisen ja helposti omaksuttavan. Kokeilumallissa käyttäjät voivat testata ohjelmistoa ilmaiseksi, mutta rajoitetun ajan ja ominaisuuksin, mikä antaa mahdollisuuden arvioida ohjelmiston soveltuvuutta ennen sen ostamista (Yang, Liu & Teo, 2017). Puolestaan pysyvässä mallissa ei ole mahdollisuutta kokeilla ohjelmistoa ennen ostamista eikä se päivity automaattisesti ilman lisäkustannuksia (Xin, 2020). Sen etuna ovat kuitenkin pysyvät käyttöoikeudet sekä muuttumaton hinta, kun ohjelmisto maksetaan etukäteen (Zhang and Seidmann, 2010).

Tilauspohjaisessa mallissa lisenssit hankitaan määräajaksi sisältäen automaattiset päivitykset. Malli mahdollistaa asiakkaille maksujen allokoimisen pidemmälle ajanjaksolle ja tarjoaa toimittajille vakaan tulovirran. Tämä malli on erityisen hyödyllinen tilanteissa, joissa ohjelmisto päivittyy usein (Xin, 2020). Palvelinmallissa lisenssikustannukset määrittyvät prosessoriytimien lukumäärän perusteella, mikä soveltuu hyvin suurten järjestelmien hallintaan. (Ferrante, 2006.)

Verkkopohjaisessa mallissa ohjelmiston asennus tapahtuu palvelimelle ja useat käyttäjät pääsevät siihen käsiksi verkon kautta. Tämä malli soveltuu ympäristöön, jossa monta käyttäjää tarvitsee pääsyn samaan ohjelmistoon. Käyttöpohjaisessa mallissa yritykset voivat hankkia lisenssejä tarpeidensa mukaan ja maksavat jokaisesta käyttökerrasta. Tämä malli yhdistyy hyvin verkkopohjaiseen malliin, sillä se mahdollistaa käyttökertojen tarkan seurannan. (Ferrante, 2006.)

Yritysten tulisi valita malli huolellisesti ottaen huomioon, miten paljon he haluavat sitouttaa pääomaa ohjelmistoon, kuinka monta käyttäjää ohjelmistolla tulee olemaan sekä mihin tarkoitukseen sitä käytetään. Ohjelmiston valinnassa tulee myös ottaa huomioon sen päivittymisen vaikutukset ja tiheys.

3 Ohjelmisto-omaisuuden hallintatyökalut suurissa yrityksissä

Ohjelmisto-omaisuuden hallintatyökalut ovat ohjelmia, jotka suorittavat ohjelmistoinventaarion hallintaa sekä keräävät tietoa ohjelmistotapahtumista (Vion ym, 2017). Ohjelmistolisensoinnin noudattamisen seuranta on esimerkki tämänlaisesta ohjelmistotapahtumasta (Paluch ym., 2018). Aikaisemmista tutkimuksista on noussut esille, että kuluttajilla on ollut vaikeuksia vertailla markkinoilla olevia ohjelmisto-omaisuuden hallintatyökaluja kustantajien harjoittaman liioitellun markkinoinnin takia. Ohjelmisto-omaisuuden hallintatyökalut ovat ideaalissa tilanteessa todella hyödyllisiä, mutta markkinoilla olevat tuotteet harvoin keräävät tarpeeksi yksityiskohtaista tietoa ohjelmistoinventaariosta. Ilman yksityiskohtaista tietoa yritykset eivät pysty tekemään perusteltuja päätöksiä ohjelmistojen poistamisesta tai seuraamaan ohjelmistojen käyttöä lisenssisopimuksissa määriteltyihin käyttöoikeuksiin. (Vion ym., 2017.) Työkalun käyttöönottaminen ei myöskään ratkaise suoraan yrityksen ongelmia. Yrityksen tulee selvittää, mitä heidän täytyy parantaa ja rakentaa sen ympärille prosesseja, joilla ongelma pystytään selvittämään (Meehan, 2002).

McCarthy ja Herger (2011) ovat muodostaneet listan, joka määrittelee vaatimukset järjestelmälle, jotta investointi olisi oikeutettu: ohjelmisto-omaisuuden näkyvyys, tunnistaa ja korjaa vaatimustenmukaisuuden noudattamatta jättämiset, tarjoaa tilastoja ja tietoa yrityksen käyttämistä ohjelmistoista, sopeutuu dynaamiseen liiketoiminta- ja teknologiaympäristöön joustavan mallin avulla sekä toimittaa palvelut kohtuullisin kustannuksin niin kehitysvaiheessa kuin jatkuvan hallinnan ja ylläpidon vaiheessa.

Vion ym. (2017) ovat esittäneet mallin, joka vertailee ohjelmisto-omaisuuden hallintatyökaluja niiden tarjoamien ominaisuuksien mukaan. Kuva 3 havainnollistaa mallin asteikon, joka koostuu neljästä tasosta: Näkyvyys, tunnistaminen, riskienhallinta ja optimointi.

1. Näkyvyyden tason saavutettuaan ohjelmisto-omaisuuden hallintatyökalu kykenee tunnistamaan jokaisen laitteen, virtuaalisen koneen ja niille allokoidut resurssit sekä jokaisen ohjelmiston, joka on asennettu mille tahansa fyysiselle tai virtuaaliselle laitteelle. Näkyvyyden puute ohjelmistojen käyttöön on yksi

suurimmista syistä ohjelmistokustannusten nousulle 20.12.2023 14.11.00(McCarthy & Herger, 2011).

2. Tunnistamisen tasossa työkalu tunnistaa ohjelmiston ja kääntää sen lisensseiksi ja tuotteiden käyttöoikeuksiksi. Tämän jälkeen käyttäjän on mahdollista tunnistaa esimerkiksi, millainen lisenssi ohjelmistolla on. Kokonaisuudessaan tämä taso auttaa ymmärretään ja dokumentoidaan, miten erilaiset ohjelmistot ovat liitettynä niiden omaisuuseriin, kuten lisensseihin ja käyttöoikeuksiin. Tämä auttaa organisaatiota ohjelmistojen hallinnassa ja niihin liittyvien tietojen käsittelyssä. (Vion ym., 2017)

3. Riskienhallinnan taso koostuu kahden ensimmäisen tason tiedon yhteen sovittamisesta. Sen tavoitteena on estää oikeudelliset ja taloudelliset riskit. Oikeudelliset riskit tulevat laittomasta ohjelmiston käyttämisestä ilman lisenssiä tai lisenssin virheellisestä käytöstä. Taloudellinen riski muodostuu lisensoitujen ohjelmistojen käyttämättä jättämisestä tai lisenssin kattavan enemmän käyttöoikeuksia kuin tarvitaan. (Vion ym., 2017.) Erityisen tärkeää riskienhallinnasta tekee ohjelmistotoimittajien suorittamien ohjelmistoauditointien lisääntyminen lähivuosien aikana (Beeman ym., 2021).

4. Optimaalisella tasolla on hyödynnetty edellisten tasojen tietoa ja kyetään tunnistamaan tapoja, joilla voidaan parantaa lisenssikustannuksia sekä tuottaa hyödyllistä liiketoimintatietoa. Lopullinen päämäärä on prosessin automatisoiminen reaaliaikaisesti. (Vion ym., 2017.)

4	Optimointi	Ajantasaisuus
3	Riskienhallinta	Monimutkaisuus
2	Tunnistaminen	Volyymi
1	Näkyvyys	Volyymi

Kuva 3. Ohjelmisto-omaisuuden hallinnan kypsyysasteikko (Vion ym., 2017)

McCarthy ja Herger (2011) ovat jakaneet työkalujen liiketoimintatavoitteet kolmeen korkean tason kategoriaan: kustannusten vähentäminen, auditointivalmius ja kolmannen osapuolten ohjelmistolisenssien käytön parantaminen.

Hankittujen ja asennettujen ohjelmistojen inventaariotietojen lisäksi, ohjelmisto-omaisuuden hallinnan tärkeimpiin hyötyihin kuuluu auditoinnin yhdenmukaisuus (Varela, Méxas & Drumond, 2018). Galimyanov ja Muzafarovan (2019) mukaan ohjelmistoauditoinnin merkittävin tavoite on löytää kaikki yrityksen koneille asennetut ohjelmistot. Jos organisaatio ovat käyttäneet ohjelmistoa luvatta ilman lisenssiä, voi se joutua maksamaan korvauksia. Esimerkiksi vuonna 2012 tietotekniikkapalveluyritys PCS-CTS joutui maksamaan 500 000 euroa Business Software Alliancelle lisensoimattomien kopioiden takia. Yleisemmin yritykset hankkivat tarvittavat lisenssit ilman minkäänlaisia oikeudellisia seurauksia. Tästä seuraa kuitenkin yleisesti lisäkustannuksia, kun tavallinen suositushinta puuttuvista lisensseistä on kaksin tai kolminkertainen listahintaan nähden. (Lawrence, Ruhl & Scott, 2014.)

Ohjelmisto-omaisuuden hallintatyökalujen käyttöönottoon liittyy myös käyttäjien kouluttaminen ja osallistaminen. Se voi muodostua haasteelliseksi erityisesti suurissa organisaatioissa, joissa on laaja käyttäjäkunta. Käyttäjien tarpeiden ymmärtäminen ja tehokas koulutus ovat avainasemassa menestyksellisessä käyttöönotossa. Ohjelmisto-omaisuuden hallintatyökalujen monimutkaisuus tuo myös mukanaan haasteita, koska niiden tehokas käyttö saattaa vaatia teknistä osaamista ja asiantuntijoiden apua. (Meehan, 2002.)

4 Yhteenveto ja johtopäätökset

Tutkielmassa selvitettiin, mitkä prosessit ja käytännöt ovat hyödyllisiä ohjelmisto-omaisuuden hallinnan optimoinnissa. Ohjelmisto-omaisuuden prosesseiksi tunnistettiin ohjelmisto-omaisuuden löytäminen, hankittujen omaisuuserien yhteensovittaminen, ohjelmistosopimustenhallinnan toteuttaminen ja liiketoimintatiedon raportointi. Prosessien lisäksi tunnistettiin neljätasoinen arviointikehys ohjelmisto-omaisuuden hallintaan, joka sisältää luotettavan tiedon tason, käytännön hallinnan tason, toiminnallisen integraation tason ja täyden ISO/IEC yhteensopivuuden tason. Tämä tutkimus käsittelee näistä luotettavan tiedon tasoa.

Tarkempaan käsittelyyn prosesseista otettiin ohjelmistoinventaarion hallinta ja ohjelmistolisensointi. Ohjelmistoinventaarion hallinnasta tutkittiin sen luominen, sen hyötyjä ja mahdollisten puutteiden vaikutuksia. Ohjelmisto-omaisuuden hallinnan tehostamiseen todettiin menetelmä hallinta tärkeyden ja poikkeuksen perusteella, joka hyödyntää ohjelmistoinventaariota. Prosessin avulla pystytään määrittämään ohjelmistojen arvollinen tärkeysjärjestys inventaarion sisällä. Tunnistettiin myös, että ohjelmistoinventaario on osa laajempaa kokonaisuutta, jonka avulla pyritään ratkaisemaan teknologian ja liiketoimintaprosessien jatkuvia muutoksia. Inventaarion ja sen sisältämän tiedon puutteet rajoittavat organisaation tietoisuutta IT-kustannuksista. Puutteet IT-kustannusten tiedoissa vaikeuttaa organisaation strategista päätöksentekoa, heikentää liiketoiminnan tavoitteiden tukemista ja aiheuttaa tehotonta resurssien käyttöä.

Ohjelmistolisensoinnista tarkasteltiin sen tärkeyttä, väärinkäytön seurauksia sekä vertailtiin eri lisensointimalleja, joita yritykset voivat hyödyntää. Tarkasteltiin myös, millaisia strategioita kuluttajat voivat hyödyntää ohjelmistolisenssin valinnassa. Tunnistettiin, että ilman voimassa olevaa lisenssiä kuluttaja ei voi käyttää ohjelmistotoimittajan tuotetta laillisesti, kun oletetaan, että kyseessä ei ole ilmaisversio. Lisenssien noudattamisen tärkeyttä korostaa, että niiden väärinkäytöstä voi seurata pahimmillaan oikeudenkäyntejä ja suuria sakkoja. Yleisesti nämä tilanteet kuitenkin selvitetään ohjelmistotoimittajan kanssa ja maksetaan tarvittavista lisensseistä markkinahintaa enemmän. Pysyvän mallin implementoimisesta saatiin selville, että yritys voi joutua suunnittelemaan uudelleen liiketoimintaprosessejaan, jotta se sopii ohjelmiston käytettävyyteen tai vaihtoehtoisesti muokkaamaan ohjelmiston ominaisuuksia, jotta ne sopivat yrityksen liiketoimintaprosesseihin.

Tutkielmassa selvitettiin myös, mitä vaikutuksia ohjelmisto-omaisuuden hallintatyökaluilla on suurille yrityksille. Lisäksi tutkielmassa tunnistettiin vaatimukset järjestelmälle, jotta siihen investointi olisi oikeutettu sekä nelitasoinen kypsyysasteikko ohjelmisto-omaisuuden hallintatyökaluille. Havaittiin, että ohjelmisto-omaisuuden hallintatyökalut teoriassa tukevat aikaisemmin käytyjä prosesseja loistavasti. Tutkielmassa kuitenkin todettiin, että markkinoilla olevat tuotteet harvoin keräävät tarpeeksi yksityiskohtaista tietoa ohjelmistoinventaariosta. Tiedon laadun tärkeys korostui niin prosessien tutkimuksessa kuin myös työkaluja tutkittaessa.

Lähteet

Albert, B.E., Dos Santos, R.P. & Werner, C.M. (2013) 'Software ecosystems governance to enable IT architecture based on software asset management', in *2013 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST). 2013 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST) - Complex Environment Engineering*, Menlo Park, CA, USA: IEEE, pp. 55–60. https://doi.org/10.1109/DEST.2013.6611329.

Beeman, A., Muchmore, J., Corey, M. & Bolton, D. (2021) *Vendor Software License Audits Have Become More Frequent. Are You Prepared?*, *Database Trends and Applications*. https://www.dbta.com/Editorial/News-Flashes/Vendor-Software-License-Audits-Have-Become-More-Frequent-Are-You-Prepared-146076.aspx (Haettu: 10 November 2023).

Ben-Menachem, M. & Marliss, G.S. (2004) 'Inventorying Information Technology Systems: Supporting the "Paradigm of Change"', *IEEE Software*, 21(05), pp. 34–43. https://doi.org/10.1109/MS.2004.1331300.

Chavarria, J.A. & Morrison, R.D. (2014) 'Revisiting Software Piracy Using Globe Cultural Practices', *Academy of Information & Management Sciences Journal*, 17(1), pp. 101–121.

Ferrante, D. (2006) 'Software Licensing Models: What's Out There?', *IT Professional*, 8(6), pp. 24–29. https://doi.org/10.1109/MITP.2006.147.

Flexera (2023) *Flexera 2023 Tech Spend Pulse | Report*. https://info.flexera.com/FLX1-REPORT-State-of-Tech-Spend (Haettu: 10 November 2023).

Galimyanov, A.F. & Muzafarova, A. (2019) 'Educational Software Audit', *Revista San Gregorio*, (32), pp. 43–50. https://doi.org/10.36097/rsan.v1i32.995.

Holsing, N.F. & Yen, D.C. (1999) 'Software asset management: Analysis, development and implementation', *Information Resources Management Journal*, 12(3), pp. 14–26. https://doi.org/10.4018/irmj.1999070102.

ISO/IEC 19770-1 (2012) *ISO/IEC 19770-1:2012(en), Information technology — Software asset management — Part 1: Processes and tiered assessment of conformance*. https://www.iso.org/obp/ui/#iso:std:iso-iec:19770:-1:ed-2:v1:en (Haettu: 6 December 2023).

Jackson, S. & Brannon, S. (2018) ‘In-house Software Development: Considerations for Implementation’, *The Journal of Academic Librarianship*, 44(6), pp. 689–691. https://doi.org/10.1016/j.acalib.2018.10.008.

Klint, P. & Verhoef, C. (2002) ‘Enabling the creation of knowledge about software assets’, *Data & Knowledge Engineering*, 41(2–3), pp. 141–158. https://doi.org/10.1016/S0169-023X(02)00038-1.

Lawrence, A., Ruhl, C. & Scott, R. (2014) ‘Surviving a Software Audit’. https://scottandscottllp.com/wp-content/uploads/2017/10/Software-Audits-Oct-2.pdf.

Li, S., Cheng, H.K., Duan, Y. & Yang, Y-C. (2017) ‘A Study of Enterprise Software Licensing Models’, *Journal of Management Information Systems*, 34(1), pp. 177–205. https://doi.org/10.1080/07421222.2017.1297636.

McCarthy, M.A. & Herger, L.M. (2011) ‘Managing Software Assets in a Global Enterprise’, in *2011 IEEE International Conference on Services Computing*. *2011 IEEE International Conference on Services Computing (SCC)*, Washington, DC, USA: IEEE, pp. 560–567. https://doi.org/10.1109/SCC.2011.119.

Meehan, M. (2002) ‘Asset Management Extends IT Control’, *Computerworld*, 36(41), p. 32.

Paluch, M., Gocek, P., Stopa, T., Kania, P. & Malecki, B(2018) ‘Obtaining Software Asset Insight by Analyzing Collected Metrics Using Analytic Services.’

Postmus, D., Wijngaard, J. & Wortmann, H. (2009) ‘An economic model to compare the profitability of pay-per-use and fixed-fee licensing’, *Information and Software Technology*, 51(3), pp. 581–588. https://doi.org/10.1016/j.infsof.2008.08.004.

Varela, A.M.Q., Méxas, M.P. & Drumond, G.M. (2018) ‘The scenario of software asset management (SAM) in large and midsize companies’, *Independent Journal of Management & Production*, 9(2), pp. 301–320. https://doi.org/10.14807/ijmp.v9i2.730.

Vion, A.-L., Baillon, N., Boyer, F. & De Palma, N. (2017) ‘Software License Optimization and Cloud Computing’, *The Eight International Conference on Cloud Computing, GRIDs, and Virtualization*, pp. 115–121.

Xin, M. (2020) ‘The Impact of Customer Valuation Uncertainty on Software Licensing’, *MIS Quarterly*, 44(2), pp. 562–603. https://doi.org/10.25300/MISQ/2020/14728.

Yang, X., Liu, N. & Teo, H.H. (2017) ‘How do users cope with trial restrictions? A field experiment on free trial software’, *International Journal of Information Management*, 37(4), pp. 339–349. https://doi.org/10.1016/j.ijinfomgt.2017.03.007.

Yuan, M., Rezaeibagha, F., Xu, L. & Huang, X. (2022) ‘Controllable software licensing system for sub-licensing’, *Journal of Information Security and Applications*, 64, p. 103061. https://doi.org/10.1016/j.jisa.2021.103061.

Zhang, J. & Seidmann, A. (2010) ‘Perpetual Versus Subscription Licensing Under Quality Uncertainty and Network Externality Effects’, *Journal of Management Information Systems*, 27(1), pp. 39–38. https://doi.org/10.2753/MIS0742-1222270103.