Mikko Tynnyrinen

Emulointi digitaalisessa pitkäaikaissäilytyksessä

Tietotekniikan kandidaatintutkielma

7. joulukuuta 2022

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Mikko Tynnyrinen

Yhteystiedot: p.mikko.i.tynnyrinen@student.jyu.fi

Ohjaaja: Jonne Itkonen

Työn nimi: Emulointi digitaalisessa pitkäaikaissäilytyksessä

Title in English: Emulation in digital preservation

Työ: Kandidaatintutkielma

Opintosuunta: Tietotekniikka

Sivumäärä: 22+0

Tiivistelmä: Tässä tutkielmassa tutkitaan kirjallisuuskartoituksen kautta emulaatiota digitaalisen pitkäaikaissäilytyksen menetelmänä. Tutkielmassa pyritään selvittämään, kuinka toimiva menetelmä emulaatio on ja mitä haasteita siihen liittyy. Emulaation teorian lisäksi tutkielmassa tutustutaan myös käytännön tutkimuksiin. Vaikka emulaatioon menetelmänä on aiemmin suhtauduttu skeptisesti, aiheesta tehty tutkimus antaa lupaavia tuloksia.

Avainsanat: emulaatio, emulaattori, digitaalinen pitkäaikaissäilytys, digitaalinen vanhentuminen

Abstract: This thesis explores emulation as a method of digital preservation through a literature review. The thesis aims to find out how effective emulation is as a method and what challenges are associated with it. In addition of the theory of emulation, the thesis also introduces practical studies. Although emulation as a method has previously been regarded with skepticism, the research conducted on the subject gives promising results.

Keywords: emulation, emulator, digital preservation, digital obsolescence

Kuviot

Sisällys

1 Johdanto

Tutkielmassa tutkitaan kirjallisuuskartoituksen keinoin kuinka tehokas ja toimiva digitaalisen pitkäaikaissäilytyksen menetelmä emulaatio on, ja mitä haasteita siihen liittyy. Emulaatio menetelmänä perustuu säilytettävän digitaalisen aineiston alkuperäisen käyttöympäristön imitointiin, jolloin alkuperäisen datan formaattia ei tarvitse muokata (Guttenbrunner ja Rauber 2012; Granger 2000; Verdegem ja van der Hoeven 2006; Rothenberg 2000b). Emulaatiota on aiemmin pidetty liian monimutkaisena ja kalliina menetelmänä, jotta sitä voisi käyttää digitaaliseen pitkäaikaissäilytykseen (Granger 2000; van der Hoeven, Lohman ja Verdegem 2007), mutta nykyään on tutkimuksia ja käytännön tuloksia, jotka ovat ristiriidassa tämän käsityksen kanssa.

Luvussa 2 käsitellään digitaalista pitkäaikaissäilytystä ja tutustutaan eräisiin digitaalisen pitkäaikaissäilytyksen menetelmiin. Luvussa 3 perehdytään emulaatioon sekä emulaattoreihin, minkä lisäksi tarkastellaan mitä haasteita ja ongelmia liittyy emulaation digitaalisen pitkäaikaissäilytyksen menetelmä sekä emulaattorien kehittämiseen. Luvussa 4 tarkastellaan käytännön kokemuksia emulaatiosta digitaalisessa pitkäaikaissäilytyksessä tutustumalla kolmeen käytännön tutkimukseen. Lopuksi luvussa 5 on tutkielman yhteenveto.

2 Digitaalinen pitkäaikaissäilytys

Digitaalinen pitkäaikaissäilytys (engl. *digital preservation*) tarkoittaa digitaalisen informaation luotettavaa, pitkäkestoista säilyttämistä. Säilytettävä aineisto voi olla joko alkujaan digitaalista tai se voi olla alun perin analogista aineistoa, joka on muutettu digitaaliseen muotoon säilyttämistä varten. Digitaalisen pitkäaikaissäilytyksen ensisijainen tarkoitus on varmistaa datan säilyvyys ja saavutettavuus, vaikka alkuperäinen aineisto tuhoutuisi tai menisi pilalle. Digitaalista pitkäaikaissäilytystä käytetään erityisesti ratkaisuna digitaaliseen vanhentumiseen (engl. *digital obsolescence*), jolla tarkoitetaan aineiston formaatin tai alkuperäisen käyttöympäristön vanhentumista siten, ettei niitä tueta nykyaikaisilla järjestelmillä (Guttenbrunner ja Rauber 2012; Granger 2000; Carta 2017; Rothenberg 2000b).

2.1 Menetelmiä

Guttenbrunner ja Rauber (2012) mainitsevat digitaalisen pitkäaikaissäilytyksen kahdeksi päämenetelmäksi migraation ja emuloinnin. Migraatiossa data muunnetaan toiseen, nykyaikaisten järjestelmien tukemaan muotoon. Migraatiota on aiemmin pidetty ainoana varteenotettavana menetelmänä suurten arkistojen säilyttämisessä (Granger 2000). Rothenberg (2000b) kuitenkin huomauttaa, että joka kerta kun datan loogista formaattia muunnetaan, on olemassa riski aineiston turmeltumisesta. Tällöin voitaisiin menettää aineiston alkuperäinen ulkoasu, rakenne, toiminnallisuus, tai pahimmassa tapauksessa varsinainen sisältö. Emuloinnissa datan alkuperäinen formaatti säilytetään, ja sen sijaan jäljitellään alkuperäistä käyttöympäristöä (Rothenberg 2000b; Guttenbrunner ja Rauber 2012). Tietokoneohjelmaa tai ohjelmistoa, joka imitoi vierasta järjestelmää tai laitetta tällä tavalla, kutsutaan emulaattoriksi. Emulointia ja emulaattoreita käsitellään tarkemmin luvussa 3.

Migraation ja emulaation lisäksi Rothenberg (1999) mainitsee vaihtoehtoisiksi lähestymistavoiksi digitaaliseen pitkäaikaissäilytykseen paperikopiot, tietokonemuseot sekä standardien mukaiset ratkaisut kuten relaatiotietokannat. Kuitenkaan mikään näistä ei Rothenbergin mukaan toimi ratkaisuna digitaaliseen pitkäaikaissäilytykseen, joskin jotkin näistä voisivat olla osa toimivaa ratkaisuna. Monia digitaalisia dokumentteja, kuten hypermediaa, ei voida jär-

kevästi tulostaa paperille lainkaan, minkä lisäksi kaikki alkuperäisen aineiston interaktiivisuus menetetään. Toisessa lähestymistavassa olisi tietokonemuseoita, joissa vanhoilla laitteilla ajettaisiin alkuperäisiä ohjelmistoja, joilla päästään käyttämään vanhentuneita dokumentteja. Tälläkin ratkaisulla on puutteita, kuten se että vanhat koneet eivät todennäköisesti pysyisi toimintakuntoisina ikuisesti, minkä lisäksi vanhentuneiden aineistojen saatavuus rajoittuisi ainoastaan näihin tiettyihin tietokonemuseoihin, joita luultavasti ei olisi kovin montaa.

Standardeihin perustuvissa ratkaisuissa digitaaliset dokumentit esitetään muodossa, jota ohjelmistot tulevat tukemaan aina. Rothenberg mainitsee esimerkkinä relaatiotietokannat. Koska kaikki relaatiotietokantojen hallintajärjestelmät toteuttavat samat toiminnallisuudet, dataa pystytään siirtämään näiden välillä ongelmitta. Ongelmia ilmenee, jos uusi malli korvaa käytetyn mallin, esimerkiksi jos oliomalli korvaisi edellä mainitun relaatiomallin. Lisäksi Rothenbergin mukaan tämä menetelmä kannustaa hallintajärjestelmien myyjiä lisäämään tuotteisiinsa standardista poikkeavia ominaisuuksia, mikä johtaisi standardien merkityksen heikkenemiseen, joskin Granger (2000) kyseenalaistaa että tämä olisi itse menetelmän vika.

3 Emulaatio

Tässä luvussa perehdytään tarkemmin emulaation sekä emulaattoreihin. Luvussa käsitellään emulaation eri käyttötarkoituksia, sekä selitetään emulaation ja virtualisoinnin ero. Emulaatioon liittyviä haasteita ja ongelmia käsitellään Grangerin (2000) ja Rothenbergin (2000b) näkökulmista, minkä lisäksi nostetaan kaksi itse emulaattorien kehittämiseen liittyvää haastetta (Takhteyev ja DuPont 2013), joista toiseen Verdegem ja van der Hoeven (2006) mainitsevat useita eri ratkaisuja.

3.1 Emulaattorit

Kuten luvussa 2 mainittiin, emulaatio menetelmänä perustuu alkuperäisen käyttöympäristön jäljittelemiseen toisella käyttöympäristöllä, jolloin säilytettävän datan alkuperäinen formaatti kyetään pitämään muuttumattomana. Tämä toteutetaan emulaattorilla, joka tyypillisesti matkii emuloitavan koneen osien toimintaa. Tällöin päästään käyttämään alkuperäisen järjestelmän käskykantaa. Yksi yleisimmistä emulaation käyttökohteista on ollut taata yhteensopivuus taaksepäin (engl. *backward compatibility*), jotta uudemmissa käyttöympäristöissä pystyttäisiin ajamaan saman valmistajan vanhemmille käyttöympäristöille kirjoitettuja ohjelmia (Rothenberg 2000b). Emulaattoria voidaan myös käyttää suunniteltavan laitteiston testaamiseen. Esimerkiksi uutta prosessoria voidaan testata emuloidussa ympäristössä ennen fyysisen laitteiston rakentamista.

Nykyään eräs tyypillinen emulaation käyttötarkoitus on vanhentuneiden ja markkinoilta poistuneiden tietokoneiden ja pelikonsolien pelien pelaaminen nykyaikaisilla tietokoneilla. Lukuisat tahot toteavatkin emulaation olevan paras lähestymistapa vanhojen videopelien säilyttämiseen (Carta 2017; Rothenberg 1999, 2000b; Guttenbrunner, Becker ja Rauber 2010). Yhteensopivuus taaksepäin ja retropelaaminen muistuttavat käyttötarkoituksiltaan digitaalista pitkäaikaissäilytystä, vaikka emulaatioon digitaalisen pitkäaikaissäilytyksen menetelmänä onkin suhtauduttu aiemmin varauksella samalla kun migraatiota on pidetty sopivimpana menetelmänä (van der Hoeven, Lohman ja Verdegem 2007).

Emulaatio muistuttaa paljolti virtualisointia, jota käytetään virtuaalikoneissa (engl. *virtual*

machine, VM). Virtuaalikoneet ovat ohjelmallisesti toteutettuja tietokoneita, joissa voidaan ajaa ohjelmia. Rothenberg (2000b) toteaa, että emulaattoria voidaan ajatella virtuaalikoneen toteutuksena ja virtuaalikonetta voidaan ajatella ohjelmana, joka emuloi kyseistä virtuaalikonetta. Olennaisin ero emuloinnin ja virtualisoinnin välillä on, että emuloidessa useimmiten käytetään emuloitavan laitteen käskykantaa, kun taas virtualisointi käyttää isäntälaitteen käskykantaa toisen järjestelmän imitointiin. Lisäksi Rothenberg mainitsee, että virtuaalikone ei yleensä vastaa mitään todellista tietokonetta, kun taas emulaattori tyypillisesti emuloi jotain tiettyä todellista tietokonetta, joka voi olla esimerkiksi historiallinen tai ehdotettu tietokone.

3.2 Haasteet ja ongelmat

Granger (2000) jakaa emulaatioon liittyvät ongelmat neljään osa-alueeseen, joita ovat tekninen lähestymistapa, sisältö, oikeudellinen ja organisatorinen sekä muut ongelmat. Tekniseen lähestymistapaan liittyy kysymys siitä mitä emuloidaan: sovelluksia, käyttöjärjestelmiä vai laitteistoalustoja. Rothenberg (2000b) toteaa laitteiston emuloinnin olevan helpompaa kuin ohjelmiston emuloinnin. Emulaattorit tyypillisesti emuloivat laitteistoa, jotta päästäisin käyttämään alkuperäistä ohjelmistoa (Guttenbrunner ja Rauber 2012). Sisältöön liittyy ongelma digitaalisen aineiston sisällön puhtaasta säilyvyydestä, mikä yleisesti pahenee digitaalisen aineiston monimutkaistumisen myötä. Tällä osa-alueella emulaatio menetelmänä on vahvimmillaan, koska alkuperäistä dataa ei muunneta, toisin kuin migraatiossa.

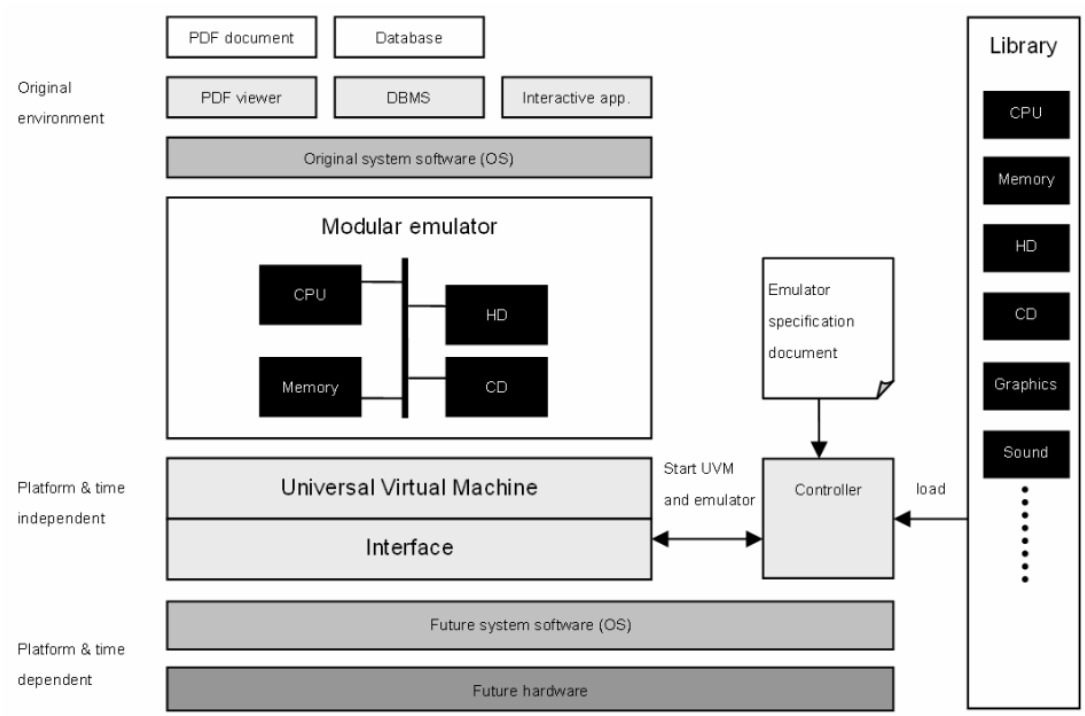
Oikeudellisiin ja organisatorisiin ongelmiin kuuluvat immateriaalioikeudet, jotka voivat liittyä ohjelmisto- ja laitteistospesifikaatioiden säilyttämiseen ja tulevaan käyttöön, vaikka ne olisivatkin vanhentuneita (Rothenberg 2000b). Lisäksi emuloinnin kustannuksien vuoksi emulaatio ei välttämättä ole vaihtoehto kaikille käyttäjille. Kuitenkin Rothenbergin mukaan vaikka emulaation kustannuksia pidetään yleisesti suurempina kuin migraation tai standardeihin perustuvien menetelmien, on tilanne todellisuudessa päinvastainen. Emuloidessa ei käytännössä tarvitse suorittaa minkäänlaista käsittelyä yksittäisiin digitaalisiin dokumentteihin, kun ne ovat hyväksytty varastoon, kun taas migraatiossa ja standardeihin perustuvissa menetelmissä vaaditaan kalliita ja toistuvia muunnoksia yksittäisille dokumenteille.

Muihin ongelmiin liittyen Granger mainitsee standardit ja avoimet spesifikaatiot, joiden

käyttöönottaminen hänen mukaansa luultavasti tekisi emulaatiosta soveltuvamman sekä kustannustehokkaamman. Lisäksi koska metatiedoilla on todennäköisesti keskeinen rooli missä tahansa emulointistrategiassa, on Grangerin mukaan tärkeä tehtävä määritellä tarkasti mitä näiden metatietojen tulisi olla ja missä muodossa. Tähän liittyen Rothenberg mainitsee, että metatiedot on tehtävä helpommin luettavaksi kuin itse kapseloidut dokumentit, jotta käyttäjän ei tarvitsisi avata kapselointia lukeakseen selityksen kapseloinnin avaamisesta.

Rothenberg (2000b) esittää joukon ratkaisemattomia kysymyksiä ja ongelmia, joista osaa on käsitelty jo edellä. Esimerkiksi pitäisikö laitteistoalustan jokainen yksittäinen malli, versio ja konfiguraatio emuloida erikseen, vai onko mahdollista emuloida yleistettyjä versioita? Kuinka tulevia käyttäjiä voidaan auttaa vanhentuneiden järjestelmien käytössä? Käyttöoppaista voidaan tarjota kansankielisiä versioita, tai tulevia niin kutsuttuja apuohjelmia voitaisiin kehittää auttamaan käyttäjiä vanhentuneiden laitteiden ja ohjelmistojen käytössä. Entä onko realistista kehittää emulaattorispesifikaatioita, jotka kuvaavat laitteistoalustoja ja -ympäristöjä riittävän yksityiskohtaisesti, jotta digitaaliset asiakirjat voidaan hahmottaa autenttisesti? Jos halutaan saavuttaa digitaalisten asiakirjojen ulkonäkö ja tuntuma täysin, spesifikaatioon voi olla tarve sisältää muun muassa tieto näytön resoluutiosta, värikalibroinnista, kuvataajuudesta sekä prosessorin ja tallennustilan nopeudesta.

Rothenberg (2000b) avaa tarkemmin laitteistoalustan yksittäisten mallien, versioiden ja konfiguraatioiden emulointiin liittyvää ongelmaa. Monessa tapauksessa sovellusohjelma on riippuvainen ainoastaan prosessorista ja yleisimmistä syöttö- ja ulostulolaitteista. Kuitenkin jotkin sovellukset hyödyntävät erityisiä komponentteja tai lisäoheislaitteita, minkä lisäksi joillain sovelluksilla voi olla erityisiä konfiguraatiovaatimuksia, kuten ylimääräistä grafiikkamuistia. Moni näistä tapauksista ratkeaisi emuloimalla yleistettyä versiota alustan konfiguraatiosta, joka sisältäisi kaikki laitteet, asetukset ja muistit, jotka olivat saatavilla kyseiselle alustalle. Siitä huolimatta jäljelle jäisi yhä tapauksia, joissa sovellus ei toimi tällaisella yleistetyllä versiolla, vaan se vaatisi toimiakseen tietynlaisen version laitteistosta. Rothenberg mainitsee yleiseksi ratkaisuksi modulaarisen emulaation, jossa yksittäiset moduulit emuloisivat yksittäisiä laitteita. Emuloitava järjestelmä voitaisiin rakentaa yhdistelemällä näitä moduuleja samaan tapaan kuin alkuperäisiä laitteita pystyi yhdistämään toisiinsa. Modulaarisen emulaation malli on esitetty kuviossa 1.



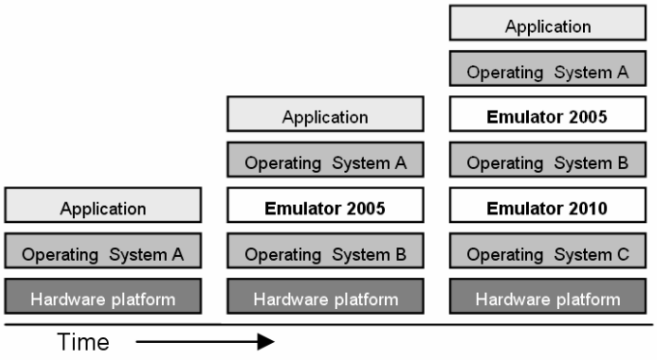
Kuvio 1. Modulaarinen emulaatio (Verdegem ja van der Hoeven 2006)

3.3 Emulaattoreiden kehittäminen

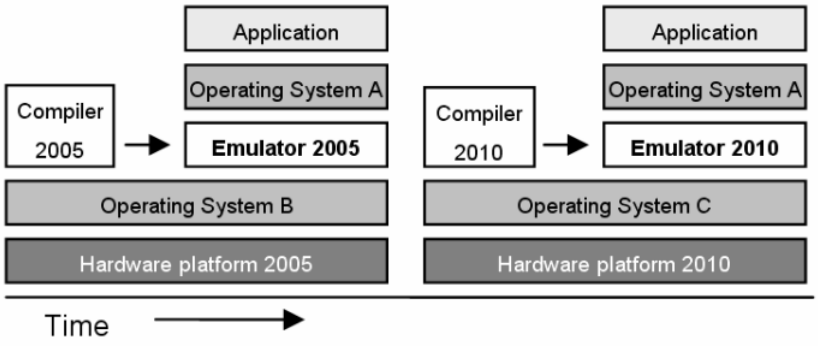
Takhteyev ja DuPont (2013) esittävät kaksi haasteta, jotka liittyvät emulaattoreiden kehittämiseen. Ensinnäkin se vaatii kehittäjältä asiantuntemusta sekä nykyaikaisista ohjelmistoalustoista että vanhoista emuloitavista järjestelmistä. Esimerkkinä Takhteyev ja DuPont antavat verkkoselaimessa ajettavan Apple II -emulaattorin, jonka kehittäminen vaatii paitsi tietämystä nykyaikaisesta JavaScriptistä, myös syvää ymmärrystä Apple II -tietokoneen toiminnasta. Sen lisäksi että emulaattorin kehittäminen on haastavaa, se ei myöskään ole koskaan valmista. Emulaattoreita itseään ajetaan alustoilla, jotka vanhenevat ennen pitkää. Aiempaa esimerkkiä jatkaen, ActiveGS-emulaattori, joka emuloi Apple II -tietokonetta verkkoselaimessa, toimi Takhteyevin ja DuPontin mukaan Firefox-selaimen vuoden 2011 versiossa, mutta ei enää vuoden 2012 versiossa.

Verdegem ja van der Hoeven (2006) mainitsevat kolme strategiaa, jotka voisivat toimia ratkaisuna emulaattorien vanhentumiseen: ketjuttaminen (engl. *chaining*), uudelleenisännöinti (engl. *rehosting, migrated emulation*) ja niin kutsuttu emulaatiovirtuaalikone eli EVM (engl.

Emulation Virtual Machine) (Rothenberg 2000a). Ketjuttamisessa jonkin tietyn tietokoneen emulaattori tarvitaan toteuttaa vain kerran. Tietokonetta, jolla emulaattoria ajetaan, voidaan tarvittaessa emuloida uudemmalla tietokoneella. Tällä tavalla voidaan toteuttaa emulaattoreiden ketju, jossa uudemmalla emulaattorilla voidaan ajaa mitä tahansa aiempaa emulaattoria. Tämä on esitetty kuviossa 2. Ketjuttaminen ei ole täysin ongelmatonta, sillä joka kerta kun alusta vanhenee, on kehitettävä uusi emulaattori tämän alustan emulointiin. Lisäksi ketjun kasvaessa kasvaa myös riski tukemattomista ominaisuuksista ja epävakaasta käyttäytymisestä.



Kuvio 2. Ketjuttaminen (Verdegem ja van der Hoeven 2006)



Kuvio 3. Uudelleenisännöinti (Verdegem ja van der Hoeven 2006)

Uudelleenisännöinnissä emulaattorin lähdekoodi kirjoitetaan nykyiselle alustalle, jossa se käännetään ajettavaksi ohjelmaksi. Kun tämä alusta vanhentuu, lähdekoodi käännetään uudestaan sellaisella kääntäjällä, jota voidaan käyttää uudella alustalla. Tämä prosessi on esitetty kuviossa 3. Emulaattori ei ole riippuvainen aiemmista emulaattoreista, mutta se vaatii

kääntäjän, joka kykenee yhdistämään vaaditut emulointitoiminnot taustalla olevaan isäntäalustaan. Kolmas Verdegemin ja van der Hoevenin mainitsema strategia on käyttää *emulaatiovirtuaalikonetta* (EVM) (Rothenberg 2000a), joka toimii välikerroksena isäntäalustan ja emulaattorin välillä. Kun jokainen emulaattori kirjoitetaan toimimaan EVM:ssä, tulee niistä alustariippumattomia. Lisäkerros kuitenkin tuo lisää monimutkaisuutta.

4 Käytännön tuloksia

Tässä luvussa käsitellään käytännön kokemuksia emulaatiosta digitaalisessa pitkäaikaissäilytyksessä tutustumalla kolmeen käytännön tutkimukseen. Ensimmäisessä tutkimuksessa kehitettiin Dioscuri-emulaattori, joka pyrkii digitaaliseen pitkäaikaissäilytykseen modulaarisen emulaation avulla (van der Hoeven, Lohman ja Verdegem 2007). Toisessa tutkimuksessa Guttenbrunner, Becker ja Rauber (2010) arvioivat eri emulaattorien ominaisuuksia videopelien pitkäaikaissäilytyksessä. Kolmannessa tutkimuksessa emuloitiin digitaalista ”The Erl King” -taideteosta (Jones 2004).

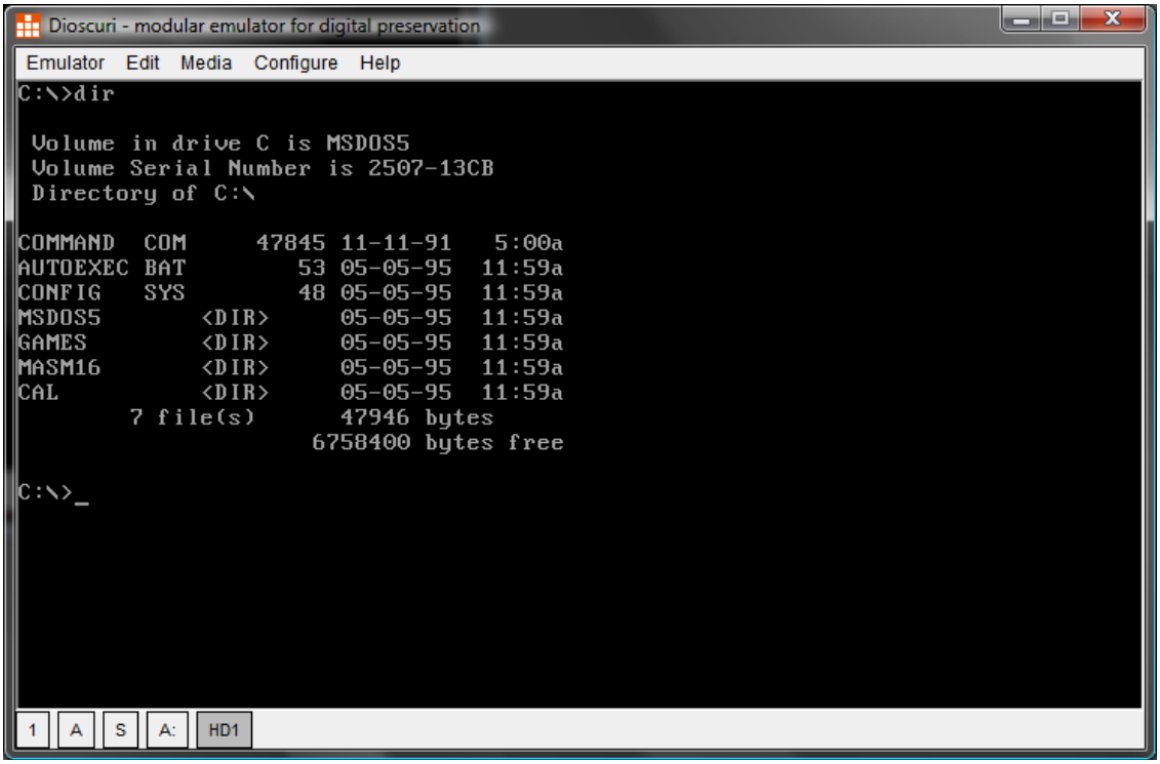
4.1 Dioscuri

Vuonna 2005 Alankomaiden kansalliskirjasto (KB) ja Alankomaiden kansallisarkisto Nationaal Archief (NA) aloittivat kahden vuoden projektin kehittääkseen emulaatioon perustuvan strategian digitaaliseen pitkäaikaissäilytykseen (van der Hoeven, Lohman ja Verdegem 2007). Projektin toteuttajien tietojen mukaan emulaatiota ei oltu koskaan aiemmin kehitetty ja testattu vastaavan kaltaisessa toimivassa digitaalisessa arkistointiympäristössä, minkä lisäksi on esitetty väitteitä että vastaavanlaisen ratkaisun kehittäminen emulaatiolla olisi liian monimutkainen ja kallis (Granger 2000). Tästä huolimatta KB ja NA uskovat, että emulaatio olisi hyvä ratkaisu digitaalisten kohteiden saatavuuden takaamiseen pitkiksi ajoiksi vaikuttamatta niiden aitouteen ja eheyteen. Projektin tavoitteena oli kehittää emulaattori, joka kykenisi luomaan uudelleen nykyaikaisen x86-tietokoneympäristön, pysyen samalla kestävänä ja helposti konfiguroitavana, minkä lisäksi emulaattorin pitäisi tukea mekanismia tietojen siirtämiseksi emuloidun ja todellisen ympäristön välillä.

Ennen emulaattorin kehittämisen aloittamista, KB toteutti alustavan tutkimuksen emulaatioon pohjautuvasta säilyttämisestä (van der Hoeven ym. 2005). Yhteistyössä Rothenbergin kanssa tämä johti uuden emulointistrategian, modulaarisen emulaation syntyyn (van der Hoeven ja van Wijngaarden 2005), jonka periaatteet perustuvat aiempiin ideoihin emulaatiovirtuaalikoneesta (Rothenberg 2000a) ja universaalista virtuaalikoneesta (Lorie 2001). Modulaarisessa emulaatiossa laitteistoarkkitehtuurin komponentteja emuloidaan yksittäisillä

emulaattoreilla, jotka yhdistämällä saadaan aikaan kokonainen emulaatioprosessi. Modulaarisen emulaation malli on esitetty luvun 3 kuviossa 1.

Emulaattorin kehitys alkoi Intel 8086 -prosessorin emuloinnilla, ja vuonna 2007 modulaarisesta emulaattorista, Dioscurista, julkaistiin ensimmäinen avoimen lähdekoodin versio. Dioscurin versio 0.2.0 kykeni ajamaan BIOS:ia sekä useita versioita MS-DOS-käyttöjärjestelmästä, josta on esitetty esimerkki kuviossa 4. Kun emulaattorin toimintaa testattiin ajamalla MS-DOS-sovelluksia sekä Dioscurilla että Windows XP:llä, huomattiin että jotkin ASCII-merkit oli korvattu toisilla merkeillä Windows XP:ssä, mikä toimi esimerkkinä siitä kuinka tietoa voitaisiin menettää ilman emulointia. MS-DOS:in lisäksi Dioscurilla kykeni ajamaan myös FreeDOS 0.9 -käyttöjärjestelmää sekä 16-bittistä Linux-käyttöjärjestelmää ELKS:iä.



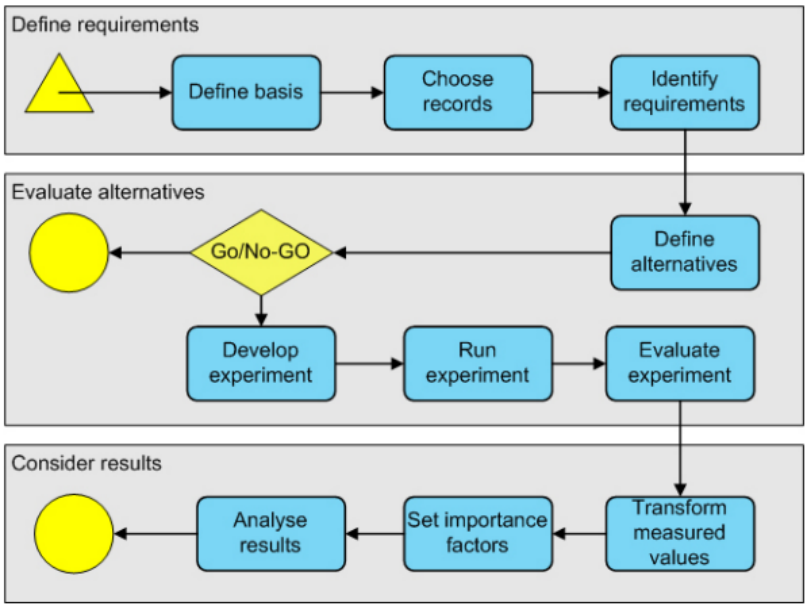
Kuvio 4. Kuvankaappaus MS-DOS 5.0:n ajosta Dioscurilla (van der Hoeven, Lohman ja Verdegem 2007)

Kahdessa vuodessa projekti osoitti, että vaikka emulaattorin kehittäminen ei ole helppo tehtävä, se on toteutettavissa, myös rajatuilla resursseilla. Artikkelin (van der Hoeven, Lohman ja Verdegem 2007) kirjoitushetkellä Dioscuri pystyi suorittamaan vanhoja sovelluksia tarkem-

min mitä nykyaikaiset tietokonealustat. Modulaarisen suunnittelun ansiosta Dioscurin voi konfiguroida emuloimaan mitä tahansa kohdealustaa riippuen saatavilla olevista moduuleista. Vuonna 2007 Dioscurista tuli osa eurooppalaista Planets-hanketta. Nelivuotisen Planets-hankkeen ensisijaisena tavoitteena oli rakentaa käytännön palveluita ja työkaluja, joilla varmistetaan digitaalisen kulttuuri- ja tiedeosaamisen pitkäaikainen käyttö. Tämän tutkielman kirjoitushetkellä Dioscurin uusin versio oli 0.7.0, joka julkaistiin vuoden 2011 tammikuussa.

4.2 Konsolipelien säilytys

Guttenbrunner, Becker ja Rauber (2010) kirjoittivat tieteellisen artikkelin, jossa he arvioivat eri strategioita digitaaliseen pitkäaikaissäilytykseen videopelien näkökulmassa, missä he keskittyvät erityisesti emulaation tutkimiseen. Artikkelissaan he kertovat heidän suorittamasta tapaustutkimuksesta, jossa he suorittivat kokeita eri emulaattoreilla, joilla emuloitiin eri videopelikonsoleita. Tutkittujen vaihtoehtojen arvioimisessa hyödynnettiin Planets-hankkeen säilyttämisen suunnittelun lähestymistapaa (Strodl ym. 2007), jonka yleiskatsaus on esitetty kuviossa 5.



Kuvio 5. Yleiskatsaus PLANETS:in säilyttämisen suunnittelun työnkulusta (Strodl ym. 2007)

Jokaista tutkimuksessa emuloitavaa pelikonsolia kohden valittiin joukko pelejä, joilla emu-

System	Alternatives	Games
Nintendo SNES	• ZSNES 1.51 • SNES9X 1.51 • MESS 0.119	• Super Mario World • Super Scope 6 • Starfox
Nintendo SNES	• video/audio grabbing • with Hauppauge WinTV PVR and viewed with VLC 0.8.6c	• Super Mario World
NEC TurboGrafx 16	• MagicEngine 1.0.0. • MESS 0.119	• Bonk's Revenge • Gates of Thunder
Sega Genesis	• Gens32 1.76 • Kega Fusion 3.51	• Sonic the Hedgehog 2 • Darxide
SNK NeoGeo	• NeoCD 0.3.1 • Nebula 2.25b	• Metal Slug • Crossed Swords 2
Coleco Telstar	• Pong 6.0 • PEmu	• Tennis
Magnavox Odyssey[2]	• O2EM 1.18 • MESS 0.119	• K.C. Munchin • Quest for the Rings
Sega MasterSystem	• Dega 1.12 • Kega Fusion 3.51	• Alex Kidd in Miracle World • Space Harrier 3D
Atari Jaguar	• Project Tempest 0.95 • MESS 0.119	• Doom • Highlander
Sony PS2	• PCSX2 0.9.2	• Gran Turismo 3 • Eye Toy Play

Kuvio 6. Tutkimukseen valitut kohdejärjestelmät, säilytettävät pelit ja vaihtoehdot pelien säilyttämiseen. (Guttenbrunner, Becker ja Rauber 2010). Kaikki vaihtoehdot käyttivät emulaatiota paitsi yksi vaihtoehto Nintendo SNES:lle, joka käytti videonauhoituksen siirtoa, sekä simulaatiota käyttävä vaihtoehto Coleco Telstarille.

laattoreita testattiin. Jokainen joukko sisälsi suositun pelin, jota todennäköisesti tultaisiin emuloimaan käytännössä, pelin joka käyttää erityistä syöttölaitetta tai peittokuvia, jotta voitaisiin testata kuinka emulaattori tukee muita kuin standardeja kontrolleja, sekä pelin joka käyttää erityistä laitteistoa tai joka julkaistiin pelikonsolin elinkaaren loppupuolella. Tämän viimeisen kaltaiset pelit käyttävät yleensä järjestelmän laitteistoa intensiivisimmin. Tutkimuksessa emuloitiin useita eri järjestelmiä, kuten Nintendo SNES, Sega Genesis ja Sony PlayStation 2 -järjestelmiä, ja jokaista järjestelmää emuloitiin usealla eri emulaattorilla. Kaikki valitut pelit, emulaattorit ja järjestelmät on listattu kuviossa 6.

Tutkimuksessa testatut emulaattorit pisteytettiin sen mukaan, miten hyvin ne toteuttivat pelien säilyttämiselle asetetut vaatimukset. Nämä vaatimukset jaettiin eri kategorioihin, jotka liittyivät esimerkiksi kohteen ominaisuuksiin (pelin visuaaliset, auditiiviset ja interaktiiviset

ominaisuudet) ja kustannuksiin. Eri kategorioille annettiin eri suuruisia painoarvoja, esimerkiksi kohteen ominaisuudet saivat suurimman painoarvon koska pelin esittäminen alkuperäisen näköisenä ja tuntuisena pidettiin korkeimpana prioriteettina.

Testeissä emulaattorien saamat pisteet vaihtelivat, myös saman kohdejärjestelmän emulaattoreiden välillä. Osa peleistä ei toiminut lainkaan osalla emulaattoreista. Tiettyjen järjestelmien emulointiin tarkoitetut emulaattorit pärjäsivät kohteen ominaisuuksien kohdalla paremmin, kun taas useiden järjestelmien emulointiin tarkoitetut emulaattorit pärjäsivät paremmin infrastruktuuriltaan. Kuviossa 7 näytetään esimerkki, kuinka tietyn järjestelmän emulointiin tarkoitettu emulaattori tuottaa ulkonäöltään paremman lopputuloksen mitä useiden järjestelmien emulointiin tarkoitettu emulaattori.

Kuvio 7. Kaksi kuvankaappausta samasta näkymästä Super Mario World -pelistä Nindento SNES:lle. (Guttenbrunner, Becker ja Rauber 2010). Vasen kuva on ZSNES 1.51 -emulaattorin tuottama, joka on tarkoitettu Nintendo SNES -järjestelmän emulointiin. Oikea, virheitä sisältävä kuva on MESS 0.119 -emulaattorin tuottama, joka on tarkoitettu usean eri järjestelmän emulointiin.

Yleisesti tutkimus osoitti, että emulaatio on toimiva strategia vanhentuneiden pelikonsolien pelien pelaamiseen nykyaikaisilla tietokoneilla. Kuitenkaan suurinta osaa tutkimuksen emulaattoreista ei voi käyttää digitaaliseen pitkäaikaissäilytykseen ilman muokkauksia, joilla pyritään parantamaan pitkäikäisyyttä. Ne emulaattorit, jotka pärjäsivät pitkäikäisyydessä paremmin, eivät jäljitelleet kohteiden ominaisuuksia riittävän hyvin.

4.3 The Erl King -tapaustutkimus

Osana Variable Media Network -ohjelmaa, Guggenheim-museo yhdessä Daniel Langlois -säätiön kanssa tutkivat tapaustutkimusten sarjaa muodostaakseen luovia strategioita uhanalaisia taideteosten säilyttämistä varten. Eräs teos, joka valittiin testaamaan emulaatiota, oli Grahame Weinbrenin ja Roberta Friedmanin interaktiivinen videoteos ”The Erl King” (1982-85) (Jones 2004; Dimitrovsky 2004). ”The Erl King” vaikutti ideaalilta kandidaatilta laitteiston emulointiin, koska siinä käytettiin vanhentunutta laitteistoa, taiteilijoiden itse kirjoittamaa ohjelmistoa sekä erikoisvalmisteisia osia. Alkuperäinen taideteos oli esillä rinnakkain emuloidun version kanssa näyttelyssä ”Seeing Double: Emulation in Theory and Practice”, jossa oli näytillä myös useita muita digitaalisia taideteoksia, ja niiden emuloituja vastineita.

Tapaustutkimus suoritettiin yhteistyössä alkuperäisten taiteilijoiden kanssa. Taideteoksen fyysiset osat eivät olleet taiteilijoille kovin tärkeitä vaan taideteoksessa käytetty ohjelmakoodi, jonka taiteilijat olivat itse kirjoittaneet yhteistyössä tietokoneohjelmoijien kanssa. Täten alkuperäisen ohjelmakoodin säilyttäminen ennallaan sai tutkimuksessa suurimman prioriteetin.

Rothenbergin oman ehdotus (Dimitrovsky 2004) ja tutkijoiden oman alustavan tutkimus olivat antaneet tutkijoiden uskoa että Internetistä löytyisi valmis emulaattori, jolla voisi emuloida teoksessa käytettyä SMC-70-tietokonetta. Kuitenkaan tutkimuksen kannalta sopivaa emulaattoria ei ollut saatavilla, eikä tapaustutkimukseen ohjelmoijan, Isaac Dimitrovskyn, mukaan ollut mielekästä kirjoittaa kokonaan uutta SMC-70-emulaattoria ottaen huomioon käytettävissä olevan ajan. Kompromissien jälkeen tutkijat löysivät ratkaisun, jolla alkuperäinen ohjelmakoodi kyettiin säilyttämään, mutta alkuperäistä SMC-70-järjestelmää ei voitu emuloida, eikä emulaatiota kyetty testaamaan kuten tutkijat olivat alun perin suunnitelleet. Tutkijat joutuivat myös miettimään muitakin kysymyksiä, kuten onko mielekästä emuloida alkuperäisessä järjestelmässä olleita virheitä, jotka aiheuttaisivat järjestelmän kaatumisen. Pilaisiko näiden virheiden poistaminen teoksen autenttisuuden? Kaikesta huolimatta tutkijat ovat sitä mieltä, että taiteilijoiden ohjauksen avulla, he onnistuivat kopioimaan taideteoksen aidon kokemuksen huolimatta sen fyysisistä komponenteista.

5 Yhteenveto

Kirjallisuuskartoituksen perusteella emulaatio on toimiva ja luotettava menetelmä digitaaliseen pitkäaikaissäilytykseen, aiempien käsityksien (Granger 2000) vastaisesti. Emulaation suurin vahvuus on ettei alkuperäistä dataa muuteta, mikä paitsi vähentää riskiä aineiston turmeltumisesta, myös vähentää menetelmän kustannuksia (Rothenberg 2000b). Vaikka emulaattoreita ei ole aiemmin käytetty tai suunniteltu digitaaliseen pitkäaikaissäilytykseen, emulaattoreiden tyypilliset käyttötarkoitukset eli vanhojen ohjelmien ja videopelien ajaminen nykyaikaisilla tietokoneilla käsittelevät samaa digitaalisen vanhenemisen ongelmaa mihin digitaalinen pitkäaikaissäilytyksin pyrkii vastaamaan. Dioscuri-emulaattori (van der Hoeven, Lohman ja Verdegem 2007) osoittaa, että rajatuillakin resursseilla on mahdollista kehittää emulaattori digitaalista pitkäaikaissäilytystä varten.

Luvussa 3 esitetyt haasteet ja ongelmat eivät ole ylitsepääsemättömiä, ja niistä moneen on esitetty ratkaisuja. Esimerkiksi laitteistoalustojen eri konfiguraatioihin liittyvään ongelmaan esitetään ratkaisuksi modulaarista emulaatiota (Rothenberg 2000b), jota Dioscurikin käyttää. Emulaattorien vanhentumiseenkin on esitetty useita ratkaisuja (Verdegem ja van der Hoeven 2006). Näitä ongelmia ja niiden mahdollisia ratkaisuja on syytä tutkia tulevaisuudessa, ja erityisesti tarvittaisiin lisää Dioscurin kaltaisia käytännön kokeita.

Lähteet

Carta, Giovanni. 2017. “Metadata and video games emulation: an effective bond to achieve authentic preservation?” *Records Management Journal* 27 (2): 192–204. https://doi.org/10.1108/RMJ-10-2016-0037.

Dimitrovsky, Isaac. 2004. “Final report, Erl-King project.”

Granger, Stewart. 2000. “Emulation as a digital preservation strategy”. *D-Lib Magazine* 6 (10).

Guttenbrunner, Mark, Christoph Becker ja Andreas Rauber. 2010. “Keeping the Game Alive: Evaluating Strategies for the Preservation of Console Video Games”. *The International Journal of Digital Curation* 5 (1): 64–90. https://doi.org/10.2218/ijdc.v5i1.144.

Guttenbrunner, Mark, ja Andreas Rauber. 2012. “A Measurement Framework for Evaluating Emulators for Digital Preservation”. *ACM Transactions on Information Systems (TOIS)* 30 (2): 1–28. https://doi.org/10.1145/2180868.2180876.

Jones, Caitlin. 2004. “Seeing Double: Emulation in Theory and Practice The Erl King Case Study”. Teoksessa *Proceedings of the Annual Meeting of the American Institute for Conservation of Historic and Artistic Work.* Portland, Oregon: Electronic Media Group.

Lorie, Raymond. 2001. “Long Term Preservation of Digital Information”. Teoksessa *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries,* 346–352. New York, NY, USA: Association for Computing Machinery.

Rothenberg, Jeff. 1999. *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation.* Council on Library / Information Resources.

———. 2000a. *An experiment in using emulation to preserve digital publications.* Koninklijke Bibliotheek.

———. 2000b. *Using Emulation to Preserve Digital Documents.* Citeseer.

Strodl, Stephan, Christoph Becker, Robert Neumayer ja Andreas Rauber. 2007. “How to Choose a Digital Preservation Strategy: Evaluating a Preservation Planning Procedure”. Teoksessa *Proceedings of the 7th ACM IEEE Joint Conference on Digital Libraries (JCDL'07),* 29–38. Vancouver, BC, Canada: Association for Computing Machinery. https://doi.org/10.1145/1255175.1255181.

Takhteyev, Yuri, ja Quinn DuPont. 2013. “Retrocomputing as preservation and remix”. *Library Hi Tech* 31 (2): 355–370. https://doi.org/10.1108/07378831311329103.

van der Hoeven, Jeffrey, Bram Lohman ja Remco Verdegem. 2007. “Emulation for Digital Preservation in Practice: The Results”. *The International Journal of Digital Curation* 2 (2): 123–132. https://doi.org/10.2218/ijdc.v2i2.35.

van der Hoeven, Jeffrey, ja Hilde van Wijngaarden. 2005. “Modular emulation as a long-term preservation strategy for digital objects”. Teoksessa *Proceedings of IWAW.* Vienna, Austria: International Web Archiving Workshop.

van der Hoeven, Jeffrey, Hilde van Wijngaarden, Remco Verdegem ja Jacqueline Slats. 2005. “Emulation – a viable preservation strategy”, https://doi.org/10.1007/11551362_47.

Verdegem, Remco, ja Jeffrey van der Hoeven. 2006. “Emulation: To be or not to be”. Teoksessa *Proceedings IS&T Archiving Conference,* 56–60. Ottawa, Canada: Society for Imaging Sciences / Technology.