TAMPEREEN
AMMATTIKORKEAKOULU

Liiketalous

TUTKINTOTYÖRAPORTTI

Ohjelmistojen virtualisointi
The virtualization of software applications

Aki Lintala

Tietojenkäsittelyn koulutusohjelma
Maaliskuu 2006
Työn ohjaaja: Harri Hakonen

TAMPERE 2006

TAMPEREEN
AMMATTIKORKEAKOULU

Tekijä(t) Aki Lintala

Koulutusohjelma(t) Tietojenkäsittely/Tietoverkkopalvelut

Tutkintotyön nimi Ohjelmistojen virtualisointi

Työn valmistumis-kuukausi ja -vuosi Maaliskuu 2006

Työn ohjaaja Harri Hakonen **Sivumäärä:** 62

TIIVISTELMÄ

Tämä tutkintotyö on kirjoitettu Atea Finland Oy:ssä käynnistyneen kehitysprojektin pohjalta, jossa tarkoituksena oli tehostaa Microsoft Windows -pohjaisten ohjelmistojen ylläpitoa sekä hallintaa koulutusluokkaympäristöissä. Itse tutkintotyö on pyritty kirjoittamaan kuitenkin yleisellä tasolla ja ympäristöriippumattomasti.

Tutkintotyössä esitellään ensin ohjelmistoylläpidon ja -hallinnan ongelmia sekä selvitetään, mistä ohjelmien asennusongelmat Windows-käyttöjärjestelmissä yleisesti johtuvat ja mistä syystä ohjelmat on asennettava työasemiin fyysisesti. Työssä esitellään erilaisia yleisimpiä tekniikoita, joilla näihin ongelmiin on pyritty vaikuttamaan. Käsiteltäviä tekniikoita ovat ohjelmistojakelutekniikka, terminaalipalvelintekniikka ja uutena vaihtoehtona ohjelmistojen virtualisointitekniikka. Työn lopuksi kuvataan esimerkkinä toteutus ohjelmistojen virtualisoinnista SoftGrid-järjestelmällä Atean kehitysprojektin pohjalta.

Työn tarkoituksena on selventää Windows-ohjelmien ydinongelmia ja herättää ajatuksia yrityksen IT-ylläpidon tehostamisesta sekä palvelulaadun parantamisesta. Tarkoituksena on myös kertoa SoftGrid-järjestelmästä, joka helpottaa IT-ylläpitäjien työrutiineja eristämällä ohjelmat itsenäisiksi säiliöiksi. Nämä ohjelmasäiliöt mahdollistavat ohjelmien jakelun käyttäjätunnus- ja lisenssiperusteisesti Windows-työasemille, -palvelimille sekä -terminaalipalvelimille ilman käyttökatkoja. Tätä tekniikkaa kutsutaan ohjelmistojen virtualisoinniksi.

Tässä työssä ei ole tarkoitus käsitellä muiden kuin Windows-käyttöjärjestelmien ohjelmistoylläpidon ongelmia, sillä Windows on hallitseva työasemakäyttöjärjestelmä maailman laajuisesti, eivätkä tässä työssä esiteltävät ongelmat ja esimerkit koske muita käyttöjärjestelmiä.

Tämä tutkintotyön perusteella voidaan todeta, että ohjelmistojen virtualisoinnista saattaa hyvin tulla hallitseva tekniikka ohjelmistojen ylläpitäjien käyttöön. Virtualisointi sopii tekniikkana sekä pienille että suurille yrityksille. Sen avulla on mahdollista päästä eroon ohjelmien fyysisistä asennuksista sekä niiden pitkäkestoisista asennustestauksista. Samalla ohjelmista saadaan täysin keskitetysti hallittavia. Nämä asiat tekevät SoftGrid-järjestelmästä tämän työn kirjoitushetkellä ehdottomasti parhaimman ohjelmistoylläpidon työkalun.

Avainsanat Windows ohjelmistoristiriidat keskitetty ohjelmistohallinta virtualisointi SoftGrid

TAMPERE
POLYTECHNIC

Author(s)	Aki Lintala
Degree Programme(s)	Business Information Systems
Title	The virtualization of software applications
Month and year	March 2006
Supervisor	Harri Hakonen

Pages: 62

ABSTRACT

This final thesis has been written on the basis of the development project started by Atea Finland Oy. In this project Atea aims to make the software support and management based on Microsoft Windows more effective in training class environment. The final thesis itself has been written in more general format and it does not depend on any environment.

At first this final thesis details some of the common problems in application deployment and management. It also explains the root cause of these problems in Windows operating systems (OS) and why, after huge development of information technology (IT), we are still required to install our applications physically in each computer. This final thesis also introduces common techniques that are used to ease these problems. Included techniques are Electronic Software Delivery, Server Based Computing and one new option Application Virtualization. In conclusion of this final thesis there is an example of the implementation of application virtualization by SoftGrid Platform based on Atea's development project.

The aim of this final thesis is to clear the picture around the core problems of Windows applications and raise thoughts about making organizations IT-support more efficient and its service quality higher. One of the aims is also to tell about SoftGrid platform that relieves administrators' work routines by isolating the applications into self-contained logical "containers". These containers make it possible to deliver pre-installed applications based on user rights and licensing to Windows workstations, servers and terminal servers without any service breaks. This technique is called application virtualization.

This final thesis has been written only for Windows OS and therefore it does not cover any other OSs' application support than Windows. All the problems and examples reported in this thesis are suitable only for Windows. This is because of the well-known fact that Windows OS is a dominating desktop operating system worldwide.

Based on this final thesis it can be stated that application virtualization has a good possibility to become the next dominating technique for organizations application support. It suites well for small companies as well for larger organizations. With application virtualization companies can get rid of physical application installations and long lasting installation testing. At the same time applications becomes centrally managed. These are the things that make the SoftGrid platform definitely the best solution for application support at this moment.

Keywords Windows application conflicts centralized application management virtualization SoftGrid

1 Johdanto

Henkilökohtaiset tietokoneet, joita nykypäivän yritysmaailmassa yleisesti kutsutaan henkilökohtaisiksi työasemiksi, on kehitetty parantamaan tiedonkäsittelyä, -saantia ja -jakamista. Ne ovat tulleet jäädäkseen. Henkilökohtaisien työasemien yleistyminen alkoi 1980-luvun alkupuolella, kun ensimmäiset helppokäyttöiset käyttöjärjestelmät sekä graafiset käyttöliittymät tulivat markkinoille, kuten esimerkiksi Mac OS, Microsoft MS-DOS ja Windows 1.0. Molempien sekä tietokoneiden että käyttöjärjestelmien kehitys ja yleistyminen on ollut nopeaa. Nykyisin henkilökohtaisia työasemia on lähes jokaisessa yrityksessä sekä kotitaloudessa.

Käyttöjärjestelmä ei yksin mahdollista henkilökohtaisen työaseman hyödyntämistä vaan lisäksi tarvitaan erilaisia ohjelmia, joilla voidaan hyödyntää käyttöjärjestelmän tarjoamia työaseman fyysisiä laitteita ja resursseja. Ohjelmat ovat työasemien käyttäjien kannalta oleellisin asia ja ne ovat olleet mukana kehityksessä aina tietokoneiden keksimisestä lähtien. Huolimatta henkilökohtaisten työasemien, käyttöjärjestelmien ja ohjelmien kehityksestä, kaikki ohjelmat on edelleen asennettava tietokoneisiin, aivan kuten 20 vuotta sitten. Olipa kyseessä sitten tavallinen henkilökohtainen Linux-, Mac OS -, Windows-työasema tai Windows Terminal -palvelin.

Yritysmaailmassa ohjelmien asennuksista vastaavat tekniset tukihenkilöt. Heidän roolinsa ei kuitenkaan lopu ohjelman asennuksen jälkeen. Nykyisin kun ohjelmat ovat kehittyneet entistä monimutkaisemmiksi ja riippuvaisemmiksi käyttöjärjestelmistä, ne vaativat myös jatkuvaa ylläpitoa ja päivitystä siinä missä käyttöjärjestelmätkin. Ohjelmien käyttöjärjestelmäriippuvuus on tuonut myös entistä enemmän ongelmia ja vianselvittelyä niin ylläpitäjille kuin kotikäyttäjillekin.

Henkilökohtaiset työasemat ovatkin nimensä ja luonteensa mukaisesti haastavia myös keskitetyn hallinnan kannalta. Nykypäivänä, kun kannettavat työasemat ja etätyöskentely ovat yleistyneet, tuovat nämä seikat lisähaasteita ylläpitäjille. Miten työasemien käyttöjärjestelmät tai ohjelmat päivitetään ja kuinka käyttäjät saisivat ohjelmia käyttöön ilman yhteyttä yrityksen lähiverkkoon?

Tosiasia on kuitenkin, että nykyisten yritysten liiketoiminnan kannalta henkilökohtaiset työasemat ovat pakollisia työntekijöille. Tukihenkilöiden rooli on pitää huolta, että käyttäjillä on käytössään heidän tarvitsemansa laitteet ja ohjelmat sekä varmistaa niiden toimivuus. Jos jokin ohjelma ei toimi tai käyttäjällä ei ole mahdollisuutta käyttää työnsä edellyttämää ohjelmaa, kärsii hänen tuottavuutensa.

1.1 Työn tarkoitus ja sisältö

Työn tarkoituksena oli tutkia mitkä seikat tekevät ohjelmistojen ylläpidosta vaikeasti hallittavan osa-alueen nykypäivän tietotekniikassa. Miksi yli 20 vuoden kehityksen jälkeen ohjelmia usein vieläkin joudutaan asentamaan erikseen tietokoneille sekä onko nykypäivänä olemassa parempaa tekniikkaa asentaa ja hallita ohjelmia helpommin, kuin mihin yleisesti on totuttu?

Työn toimeksiantaja

Atea Finland Oy on IT-tuotteita, IT-ratkaisuja ja IT-palveluja tarjoava pohjoismainen yritys, jolla on Suomessa 11 toimipistettä, oma logistiikkakeskus ja palveluksessa noin 300 henkilöä. Atea on toiminut Suomessa 20 vuotta. (http://www.atea.fi).

Kehitysprojekti

Atea Finland Oy ylläpitää ja kehittää erään asiakkaansa koulutusluokkaympäristöjä. Atea on käynnistänyt koulutusluokkien osalta uuden kehitysprojektin, jossa tarkoituksena on tehostaa ohjelmien asennustekniikkaa ja hallintaa koulutusluokkaympäristöissä. Lisäksi tavoitteena on saada yhtenäisempiä ohjelmaympäristöjä, jotta kouluttajille ja oppilaille voidaan taata aina samanlainen toiminnallisuus, koulutusluokasta riippumatta. Projektin tavoitteet pyritään toteuttamaan hyödyntämällä Softricityn kehittämää SoftGrid-järjestelmää, jolla voidaan eristää ohjelmat toisistaan sekä käyttöjärjestelmästä, itsenäisiksi ja käyttöjärjestelmästä riippumattomiksi virtuaalisiksi ohjelmiksi. Tämä tutkintotyöraportti on kirjoitettu tämän kehitysprojektin pohjalta.

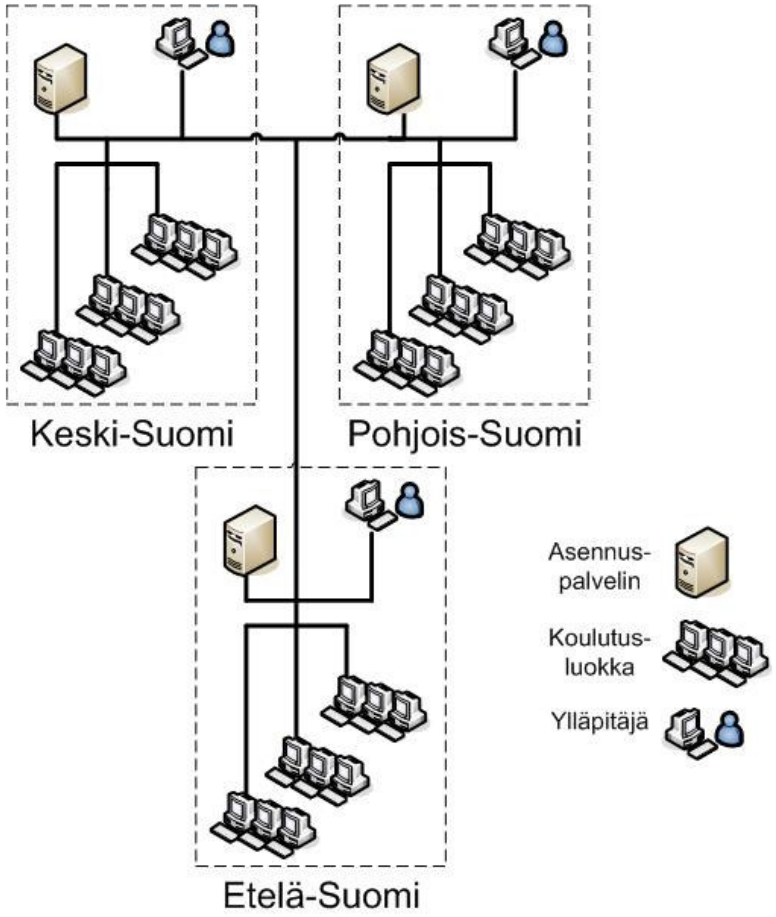
Tutkintotyön sisältö

Tutkintotyön sisältö koostuu teoriaosuudesta ja käytännön osuudesta. Luvussa kolme kerrotaan ohjelmistojen ylläpidon ongelmista. Luvussa neljä käsitellään erilaisia nykyisiä toimintamenetelmiä sekä esitellään virtualisointitekniikka uutena tapana lähestyä nykyisiä ongelmia. Luvussa viisi teoriaosuuden lopuksi esitellään miten SoftGrid-järjestelmällä voidaan ongelmiin vaikuttaa ja miten ohjelmien virtualisointi ja hallinta on tässä järjestelmässä toteutettu.

Tutkintotyön käytännön osuudessa luvussa kuusi kuvataan Atea Finland Oy:n ylläpidon alaisuudessa olevien koulutusluokkien ohjelmistohallinnan kehittämisprojektia. Luvussa kerrotaan tarkemmin, minkälainen ympäristö on kyseessä sekä mitä asioita SoftGrid-järjestelmällä odotetaan saavutettavan. Lisäksi luvussa kuusi kerrotaan projektin tämän hetkinen tilanne, mitä ongelmia tai rajoitteita SoftGrid -järjestelmässä on ja mitä sillä on saatu parannettua.

1.2 Työn toteutusympäristö

Koska käytännön työ on toteutettu Atean asiakkaalle, kuvataan tässä työssä lähinnä Atean heille toteuttamaa koulutusluokkaympäristöä ja sen kehittämistä. Koulutusluokkaympäristö koostuu eri alueista, jotka jakaantuvat seuraavasti (Kuva 1.1): Etelä-Suomi, Keski-Suomi ja Pohjois-Suomi. Jokaisella alueella on omat ylläpitäjänsä ja heidän vastuullaan on ensisijaisesti oman alueensa koulutusluokat, joita on keskimäärin seitsemän jokaista kolmea aluetta kohti. Koulutusluokat eivät kuitenkaan sijaitse eri alueilla samoissa rakennuksissa, vaan ne sijaitsevat osittain eri kaupungeissa. Tästä syystä kaikki asennukset luokkien työasemille tehdään pääsääntöisesti etäasennuksina. Lähtötilanteen järjestelmällä on ylläpitäjillä mahdollisuus myös tehdä etäasennuksia kaikille muillekin alueille ja helpottaa näin toistensa työkuormaa. Hallittavia työasemia luokkaympäristössä on yhteensä noin 300.



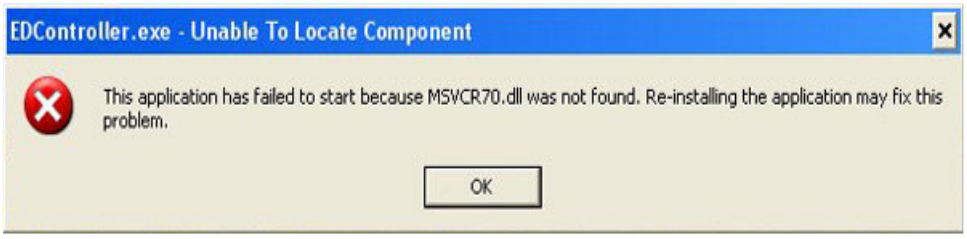
Kuva 1.1: Lähtötilanteen koulutusluokkaympäristö.

2 Ohjelmistoylläpidon yleiset ongelmat

Useimmissa yrityksissä eniten huomiota keräävät työasemien laiteviat ja vähiten huomiota saavat ohjelmistojen aiheuttamat ongelmat, sillä työasemien fyysiset laiteviat ovat helpompia käsitellä hallinnon näkökulmasta. Niihin voidaan osittain varautua budjetissa sekä ostamalla varastoon valmiiksi kuluvia osia, esimerkiksi kovalevyjä.

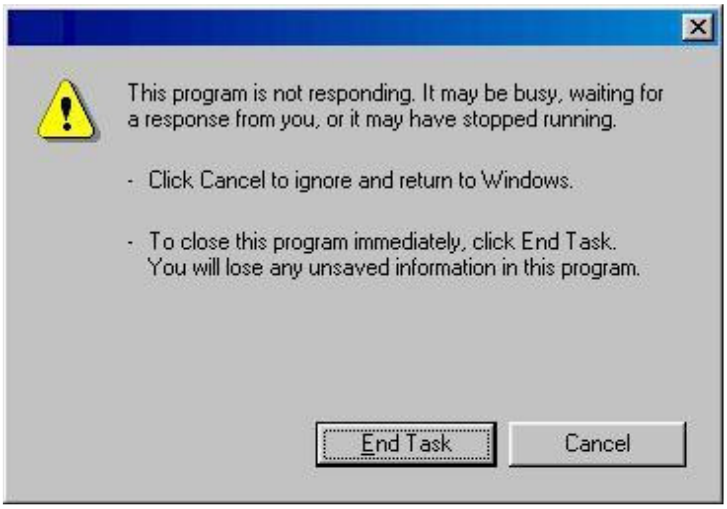
Ohjelmistoihin liittyviin ongelmiin on vaikeampaa ennalta varautua. Ohjelmat eivät oireile ongelmista etukäteen ja usein virheiden ilmaantuessa ongelman aiheuttanut muutos on tapahtunut niin paljon aikaisemmin, ettei syytä siihen enää osata yhdistää (Kuva 2.1).

Kuitenkin useille tietokoneen käyttäjille ovat seuraavat virheilmoitukset valitettavan tuttuja:
”*Ohjelma on suorittanut laittoman toiminnon ja se lopetetaan*”
”*Ohjelma X aiheutti sivuvirheen moduulissa ABC.dll*”
”*Ohjelma X aiheutti poikkeuksen c0000006h moduulissa ABC.dll*”



Kuva 2.1 Esimerkki virheilmoituksesta.

Lisäksi on tilanteita, joissa virheilmoituksia ei synny lainkaan. Tietokone ”jumiutuu” tai ohjelman ikkunaan tulee ilmoitus, ettei ohjelma vastaa, jolloin käyttäjä joutuu sen sammuttamaan väkisin (Kuva 2.2). Sammutettaessa ohjelma väkisin, käyttäjä yleensä menettää myös tekemänsä työn tai ainakin sen osan, jota ei ollut talletettu.



Kuva 2.2 Esimerkki jumiutuneesta ohjelmasta.

Näiden ilmoitusten perusteella käyttäjät harvoin tietävät tai saavat selville, mikä tietokoneessa todella on vialla. Sama koskee ylläpitäjiä. Heidän on selvitettävä onko virhe yleinen vai yksittäinen, johtuuko virhe jostakin käyttöjärjestelmän osasta vai itse asennettavan ohjelman osasta, onko mahdollisesti joku toinen havainnut samanlaisen ongelman ja löytänyt siihen ratkaisun.

2.1 Ongelmien lähtökohdat

Historiallisesti ohjelmistoylläpidon ongelmien lähtökohtana voidaan pitää henkilökohtaisten työasemien keksimistä ja toisaalta sitä hetkeä kymmenen vuotta sitten, kun Microsoft julkaisi uuden Windows 95 -version suositusta käyttöjärjestelmästään. (Brown 2005: 2.)

Henkilökohtaiset työasemat toivat mukanaan ohjelmistojen hallittavuusongelmat. Henkilökohtaiset työasemat voivat sijaita missä tahansa yrityksen lähiverkossa ja kannettavien työasemien yleistymisen myötä ne voivat sijaita missä päin maailmaa tahansa. Yrityksille syntyi tarve keskitettyyn hallintaan. Tapaan, jolla voitaisiin keskitetysti hallita ohjelmistojen elinkaari niiden asennuksesta ja päivityksestä aina poistamiseen asti. (Brown 2005: 2.)

Windows 95 -käyttöjärjestelmä toi mukanaan uudet käsitteet kuten rekisteri ja dynaamisesti linkitetyt kirjastot (DLL-tiedostot), jotka ovat edelleen käytössä kaikissa Windows-tuotteissa. Nämä muutokset otettiin käyttöön, jotta ohjelmat sekä käyttöjärjestelmä voisivat kommunikoida ja jakaa tietoa paremmin toisilleen. Uudet muutokset toivat kuitenkin mukanaan myös uudet ongelmat. Jokaisen uuden ohjelman asennus aiheuttaa riskin rekisteri- ja komponenttiristiriidoille. (Brown 2005: 2.)

Ennen Windows 95 -aikaa, ohjelmistot olivat hyvin itsenäisiä kokonaisuuksia. Ohjelmat koostuivat usein vain yhdestä käynnistystiedostosta ja muutamista konfiguraatio-tiedostoista. Ylläpitäjät pystyivät helposti asentamaan saman ohjelman useampaan työasemaan kopioimalla vain sen tiedostot työasemalta toiselle. (Brown 2005: 2.)

Henkilökohtaiset työasemat ja käyttöjärjestelmien kehitys ovat siis tuoneet mukanaan kaksi suurta ongelmaa ohjelmistojen ylläpitoon:

1. Käyttöjärjestelmän ominaisuuksista johtuen ohjelmistoasennukset aiheuttavat vaikeasti korjattavien virheiden uhan.

2. Ohjelmilta puuttuu niiden keskitetty hallinta koko elinkaaren ajaksi.

2.2 Ohjelmistojen asennusongelmat

Ongelman ydin löytyy ohjelmien nykyisestä asennustavasta. Ne saattavat pysyvästi muuttaa käyttöjärjestelmää. Ohjelma-asennukset pystyvät lisäämään omia asetuksiaan käyttöjärjestelmän rekisteriin. Ne pystyvät myös asentamaan oman version dll-tiedostoista käyttöjärjestelmään korvaten näin jo olemassa olevan dll-tiedoston. Joissain tapauksissa voi tapahtua molemmat. Tämä lopulta johtaa konflikteihin ja versioristiriitoihin toisten ohjelmien kanssa. Pahimmillaan ristiriidat estävät useimpien ohjelmien tai jopa koko käyttöjärjestelmän käytön. Tämä työllistää ylläpitäjiä heidän etsiessään ongelman aiheuttajaa ja ratkaisua ristiriitoihin. (Brown 2005: 3.)

Esimerkki asennuksessa syntyvästä mahdollisesta ristiriidasta (Pratschner 2001):

1. Käyttäjä asentaa Microsoft ActiveX Control -komponentin vierailessaan jollakin Web-sivulla.

2. Asennus korvaa työasemassa olevan vanhemman version kyseisestä komponentista.

3. Mikäli työasemaan on aiemmin asennettu ohjelma, joka tarvitsee vanhemman version Control-komponentista, ei ohjelma enää käynnisty tai toimi niin kuin sen pitäisi.

Asennuksessa ongelmia aiheuttavat yleisimmin:

- **Versiointi** – Useat ohjelmat käyttävät samoja jaettuja komponentteja kuten dynaamisesti linkitettyä kirjastoa (*DLL – dynamic-linked library*) tai COM-komponentteja (*COM – Component Object Model*). Tyypillisin tilanne on, että asennus asentaa uuden version jaetusta komponentista, joka ei ole taaksepäin yhteensopiva. Vaikka juuri asennettu ohjelma toimiikin, estää sen asennus muiden ohjel-

mien toiminnan, jotka käyttävät aikaisempaa versiota jaetusta komponentista. (Pratschner 2001)

- **Rekisteröinti** – Ohjelman asennus vaatii monenlaisien jaettujen komponenttien kopioimista kovalevylle. Jotta käyttöjärjestelmä osaisi niitä käsitellä, vaativat kaikki nämä jaetut komponentit omat merkintänsä käyttöjärjestelmän rekisteriin. Tästä toimintatavasta johtuen on ohjelmien poistaminen puhtaasti, jälkeäkään jättämättä lähes mahdotonta, koska rekisterimerkinnöistä harvemmin pidetään kirjaa. Tämä asia myös hankaloittaa ohjelmien uudelleenpaketointia keskitettyä ohjelmistojakelua varten. Ohjelmien poisto käyttöjärjestelmästä on osittain jätetty myös ohjelmistovalmistajien huoleksi. Toiset heistä hoitavat asiansa paremmin kuin toiset. (Pratschner 2001)

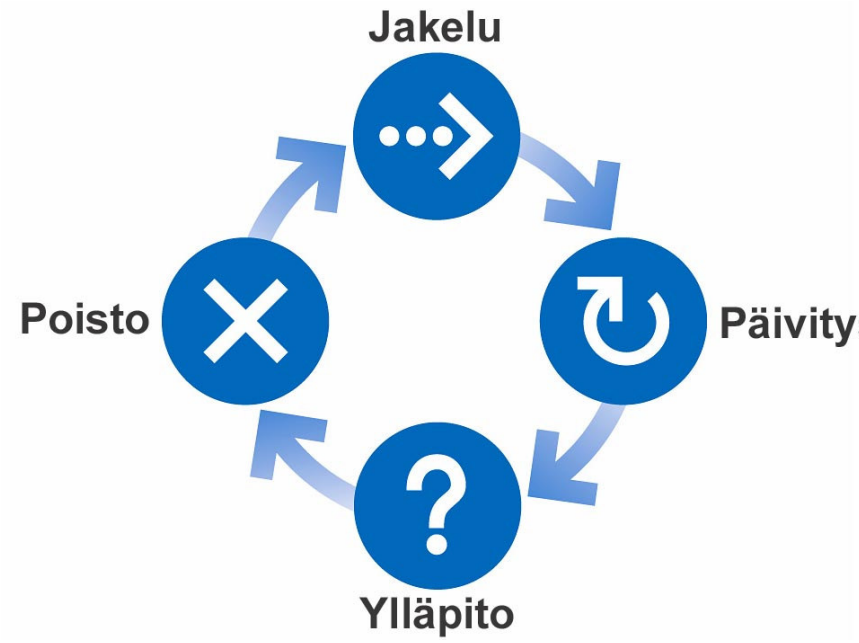
- **Ohjelman eri versiot** – Versiointi- ja rekisteröintitavoista johtuen ei yleensä ole mahdollista myöskään asentaa useampaa eri versiota samasta ohjelmasta, koska viimeisimmän version asennus korvaisi aikaisempien versioiden käyttämät jaetut komponentit. Nykyisin ohjelmien asennukset huomioivat aikaisemmat asennetut versiot samasta ohjelmasta ja osaavat päivittää tai korvata ne, mutta samanaikaisesti kahta eri versiota samasta ohjelmasta ei silti voida käyttää.

Ohjelmien asentamat jaetut komponentit pitää siis erikseen kuvailla rekisterissä ja niiden versioista ei ole pakotettua kirjanpitoa. Toisin sanoen ohjelmat eivät ole itsenäisiä kokonaisuuksia, vaan ne ovat riippuvaisia käyttöjärjestelmästä, johon ne ovat asennettuina.

2.3 Ohjelmistojen hallintaongelmat

Käyttöjärjestelmän jaetut komponentit, rekisteriominaisuus ja henkilökohtaisten työasemien yleistyminen eivät ole tuoneet ongelmia pelkästään ohjelmien asennukseen vaan myös ohjelmien hallintaan. Koska ohjelmat ovat sidoksissa käyttöjärjestelmään, johon ne ovat asennettuina, ei niitä voida vain kopioida tai siirtää tietokoneelta toiselle.

Sama koskee ohjelmien poistoa. Ohjelmia ei voida poistaa tuhoamalla vain sen kansio ja tiedostot. Ohjelmien poistoa varten on olemassa omat poisto-ohjelmat, jotka poistavat ohjelman tiedostot, mutta pääsääntöisesti ne jättävät jaetut komponentit sekä niiden rekisterimerkinnät tietokoneille. Steven Pratschnerin (Pratschner 2001) mukaan tämä johtuu siitä, että ohjelmien jaetut komponentit sijaitsevat erillään niiden rekisterimerkinnöistä. Toisin sanoen ohjelmien poisto-ohjelmat eivät tiedä minne kaikkialle poistettavan ohjelman komponentit on esitelty rekisterissä ja käyttääkö mahdollisesti jokin toinen ohjelma samoja jaettuja komponentteja.



Kuva 2.3 Ohjelmistojen hallintaprosessi. Lähde: www.softricity.com

Kuvassa 2.3 kuvataan ohjelmistojen hallintaprosessia yksinkertaisimmillaan. Mikäli prosessin jokaiseen kohtaan pystyttäisiin lisäämään ensimmäiseksi sana ”*keskitetty*”, saataisiin ohjelmistojen hallinnasta huomattavasti nykyistä tehokkaampaa. Vaikka henkilökohtaiset työasemat ovatkin yhteydessä toisiinsa lähiverkon avulla, vaativat ne silti ohjelman fyysisen asennuksen, päivityksen, ylläpidon ja poiston yksilöllisesti jokaiselta tietokoneelta. Suurissa yrityksissä tietokoneet voivat sijaita kaukanakin toisistaan, joten ohjelmien ylläpidosta syntyy näin aikaa vievää työtä.

2.4 Tutkimus tietotekniikan ongelmista

Cap Gemini Ernst & Young (CGE&Y) ja Tietotekniikan Liitto (TTL) teettivät tutkimuksen Suomen Gallupilla (TNS Gallup Oy) kesällä 2003 aiheesta ”*Suomalaisten työntekijöiden tietokoneen käytön ongelmia suomalaisessa työelämässä*.”. Tutkimuksessa haastateltiin 500 päivittäin työssään tietokonetta käyttävää ihmistä (Kuva 2.4). (Tietoa 2003: 6-8.)

Kuva 2.4: Kuinka paljon ongelmat kuluttavat työaikaa viikosta minuutteina. Lähde Tietoa 2003: 6-8.

Tutkimuksen mukaan ohjelmat aiheuttavat käyttäjille säännöllisesti ongelmia. ”*Keskimääräiseltä käyttäjältä kuluu yli 60 minuuttia viikossa sovelluksien ongelmatilanteisiin*”. Suurimpana yksittäisenä kohtana oli toisten auttaminen ongelmatilanteissa johon aikaa kului 52 minuuttia työviikosta. Eli kun ongelmia ilmenee, pyytää käyttäjä apua ensimmäisenä työkaveriltaan, jolloin hänenkin työnsä keskeytyy ja molempien aikaa kuluu ongelmien selvittelyyn. Osa tästä ajankäytöstä johtuu ohjelmistojen puutteellisesta koulutuksesta ja osa taas laitteistojen hajoamisesta. Laitteen toimimattomuudesta johtuvat ongelmat ovat helposti todettavissa. Osa ongelmista johtuu taas itse ohjelmista ja nämä ongelmat ovat hankalampia selvittää. Esimerkiksi jos tiedostot eivät aukea ohjelmissa tai jokin ohjelma ei vain tee mitä sen pitäisi tehdä. Ensin tätä ongelmaa selvittelevät käyttäjät työkavereineen ja lopuksi vielä tekniset tukihenkilöt tai ylläpitäjät. (Tietoa 2003: 6-8.)

3 Miten ongelmia on pyritty ratkaisemaan?

Alkuvaiheessa ylläpitäjät yrittivät ratkaista asennusongelmia antamalla käyttäjille useamman tietokoneen, yhden ohjelmalle X ja toisen ohjelmalle Y. Tämä ei tuonut ratkaisua ongelmaan vaan lisäsi ylläpitokuluja sekä hankaloitti hallintaa entisestään. Mikäli kyseessä olisi ollut ympäristö, jossa esimerkiksi viisi eri ohjelmaa ei toimisi samassa tietokoneessa, tulisi kuluista aivan liian suuria tällä menetelmällä. (Brown 2005: 4.) Tähän asti paras tulos on saavutettu *regressio-testauksella* eli testaamalla jokainen ohjelma toistensa kanssa kaikilla yrityksen työasemalaitteistokokoonpanoilla. (Softricity DoD 2005: 1-2.)

Tämä menetelmä on usein kuitenkin yritysten kannalta niin kallista ja aikaa vievää, että kaikkia vaihtoehtoja ohjelmistoista ja työasemista ei voida testata. Lopuksi varsinaisen testauksen suorittavat itse tietokoneiden käyttäjät. Mikäli ongelmia ilmenee, pitää ylläpitäjän käydä paikan päällä ratkaisemassa niitä.

Nykyisin monet yritykset ovat lähteneet yhtenäistämään eli vakioimaan työasemia, niiden oheislaitteita sekä ohjelmistoja helpottaakseen ohjelmien asennusta ja varsinkin testausta. Tietokoneiden, käyttöjärjestelmien ja ohjelmien jatkuva kehitys vanhentaa niitä nopeasti. Tietokoneiden mallit saattavat vaihtua useasti jo vuodessa. Lisäksi osa ohjelmista on laitteistoriippuvaisia, kuten esimerkiksi CAD-suunnittelu-ohjelmat, jotka vaativat tehokkaan erikoisnäytönohjaimen ja paljon keskusmuistia. Nämä asiat hankaloittavat työasemien vakiointia, koska kaikkien koneiden ei tarvitse olla yhtä tehokkaita.

Jotkut yritykset ovat ottaneet käyttöön ohjelmien tarjonnan käyttäjille yrityksen sisäisessä verkossa, eli käyttäjä saa itse valita haluamansa ohjelman tai ohjelmat, jotka asentuvat automaattisesti käyttäjän pyynnöstä työasemaan. Näin saadaan ohjelmien asetukset ja versiot pysymään ainakin jossain määrin yhtenäisinä. Yhtenä esimerkkinä tällaisen järjestelmästä on IBM:n kehittämä Tivoli-järjestelmä.

3.1 Ohjelmistojakelutekniikka

ESD-tekniikka (*Electronic Software Delivery*) on yleisin ohjelmistojen hallintajärjestelmä, jolla ylläpitäjät voivat asentaa ohjelmapaketteja usealle tietokoneelle massa-asennuksena hyödyntämällä verkkoyhteyksiä.

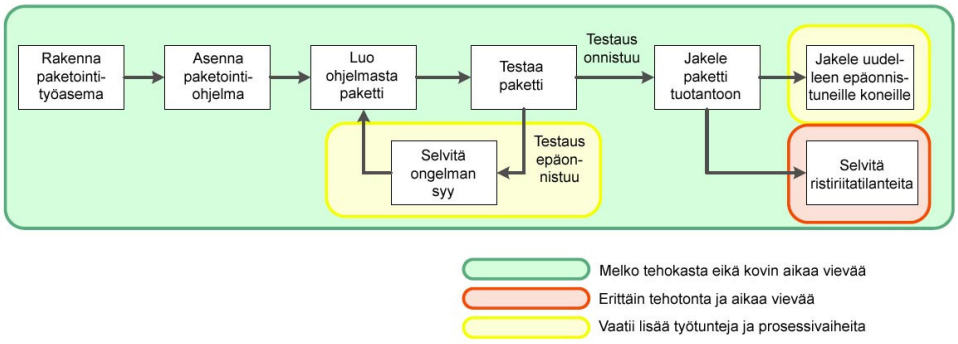
Yleisimpiä ESD-järjestelmiä ovat mm. Microsoft Systems Management Server, Altiris, Novell ZenWorks, Citrix Installation Manager ja IBM Tivoli.

3.1.1 ESD-tekniikan toimintamalli

Luodakseen jaeltavia ohjelmapaketteja, ylläpitäjän on rakennettava ja asennettava useita erilaisia testausympäristöjä, jotka vastaavat niitä työasemaympäristöjä, joihin ohjelmapaketteja on ajateltu jakaa. Tämä on osa regressio-testausta, jotta ohjelmapaketteja jaeltaessa ei syntyisi ongelmia. (Brown 2005: 6-7.)

Kun testausympäristö on saatu valmiiksi, asennetaan ympäristön työasemille *paketointiohjelma*, jonka avulla halutuista ohjelmista voidaan tehdä jaeltavia paketteja. Tämä työkalu toimitetaan ESD-järjestelmien mukana. Paketoitava ohjelma asennetaan normaalin käytännön mukaisesti työasemaan. Samalla paketointityökalu tarkkailee taustalla ohjelma-asennuksen tekemät muutokset työasemaan ja luo niiden perusteella asennetusta ohjelmasta oman jaeltavan ohjelmapaketin. (Brown 2005: 6-7.)

Tämän jälkeen luotu ohjelmapaketti testataan, ettei se aiheuta ongelmia jaeltaessa sitä yrityksen työasemaympäristössä. Mikäli paketti läpäisee testauksen, voidaan sen jakelu aloittaa yleisesti. (Brown 2005: 6-7.)



Kuva 3.1 ESD-tekniikan toimintakuva. Lähde: Brown 2005: 7.

Kun tutkitaan kuvan 3.1 perusteella ESD-tekniikan prosessia ajankäytön kannalta, löytyy prosessista monia aikaa kuluttavia ja tarpeettomia kohtia. Nämä tekevät ESD-tekniikasta hankalan, mutta silti paremman kuin manuaalisesta asennuksesta. Ongelmallisinta ESD-tekniikassa on se, jos ohjelmapaketin luonti ei onnistukaan suunnitellusti. Lähes jokaisessa ohjelmapaketin asennuksessa ilmenee tietty määrä ongelmia. Mitä monimutkaisemmaksi ohjelmapaketit menevät, sitä enemmän ongelmia niiden asennuksessa ilmenee. (Brown 2005: 7.)

3.1.2 Ohjelmistojen hallintaprosessi ESD-tekniikalla

Ohjelmien asennus, ylläpito, päivitys ja poisto ESD-tekniikalla (Brown 2005: 8):

- **Asennus** – Testausympäristön rakennus, jaeltavan paketin luonti ohjelmasta ja ohjelmapaketin regressio-testaus tunnetuilla kokoonpanoilla. Kun ohjelmapaketti läpäisee testauksen, voidaan se jaella käyttäjien työasemille, jossa ohjelmaa käytettään paikallisesti.

- **Ylläpito** – Mikäli ongelmia ilmenee, ylläpitäjän on käytävä paikanpäällä niitä selvittämässä joko etäyhteydellä tai fyysisesti.

- **Päivitys** – Pelkkien päivitysten paketointi jaeltaviksi paketeiksi onnistuu ja ne voidaan asentaa suoraan vanhan ohjelman päälle. Mutta jos tämä epäonnistuu, on ohjelmasta luotava uusi jaeltava ohjelmapaketti, jossa on mukana päivitykset. Nämä kaikki on myös testattava ja jaeltava onnistuneesti uudelleen.

- **Poisto** – Ohjelmien poisto onnistuu vain paikallisesti ohjauspaneelista Lisää/Poista sovellus -painikkeella tai ohjelmaskriptillä, joka käynnistää ohjelman poistotoiminnon. Nämä tavat jättävät kuitenkin käyttöjärjestelmään usein tarpeettomia jaettuja ohjelmakomponentteja ja rekisterimerkintöjä.

3.1.3 ESD-tekniikan hyvät ja huonot puolet

Hyvät puolet:

- Manuaaliseen asennukseen verrattuna, ESD-tekniikassa voidaan asentaa keskitetysti ohjelmia massa-asennuksina.

- Ohjelmat ja käyttöjärjestelmät voidaan päivittää keskitetysti.

- Useimmilla ESD-järjestelmillä on mahdollista ylläpitää ja hallita laiterekisteriä.

Huonot puolet:

- Ohjelmat asennetaan fyysisesti työasemiin. Kuten aiemmin todettiin, fyysiset ohjelma-asennukset aiheuttavat ongelmia.

- Tukihenkilöiden täytyy virhe- ja tukitilanteissa aina käydä tietokoneella joko fyysisesti tai etähallintaohjelmalla.

- Ohjelmia ei voida poistaa siististi ESD-tekniikan avulla ja näin tietokoneelle jää ylimääräisiä tiedostoja sekä rekisterimerkintöjä, jotka ajan myötä aiheuttavat ongelmia.

- Kaikki työasemat, joihin ohjelmia ESD-tekniikalla asennetaan, ovat pois käytöstä ohjelmien asennuksen ajan, mikäli asennukset tehdään

työaikana. Mikäli asennukset tehdään illalla tai yöllä, on tämä ajan kohta hankala taas ylläpidon kannalta, sillä ihmiset ovat saattaneet sulkea työasemansa tai viedä kannettavat työasemat mukanaan.

3.2 Server-Based Computing -tekniikka

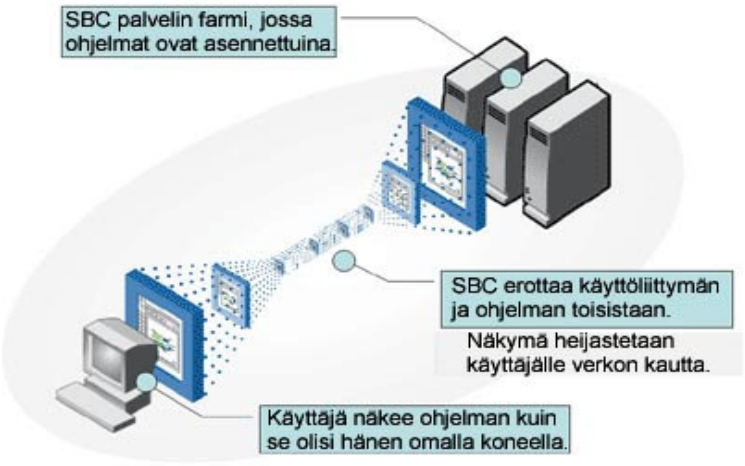
SBC-tekniikalla käyttäjät pääsevät käyttämään ohjelmia palvelimelta, aivan kuin ne olisivat asennettuina heidän omalle tietokoneelleen. Toisin sanoen ohjelmat ovat asennettuina palvelimella, mutta niiden näkymä heijastetaan käyttäjän näytölle. (Brown 2005: 8-9.)

Yleisimmät SBC-järjestelmät ovat Citrix Presentation Server ja Microsoft Terminal Server. (Brown 2005: 8.)

Server-Based Computing -tekniikka mahdollistaa keskitetyn ohjelmistojen hallinnan. Tämä tekniikka koostuu yleensä useammista palvelimista, joita kutsutaan *farmiksi* (Brown 2005: 9). Farmin palvelimet ovat yhdistettyinä toisiinsa kuormantasaustekniikalla (Load-Balancing) (Madden 2004). Tällä saavutetaan palvelinten kuormituksen jako sekä parempi vikasietoisuus. Mikäli jokin farmin palvelimista hajoaa tai se joudutaan sammuttamaan esimerkiksi päivitysten asentamisen vuoksi, pitävät farmin toiset palvelimet palvelua edelleen yllä.

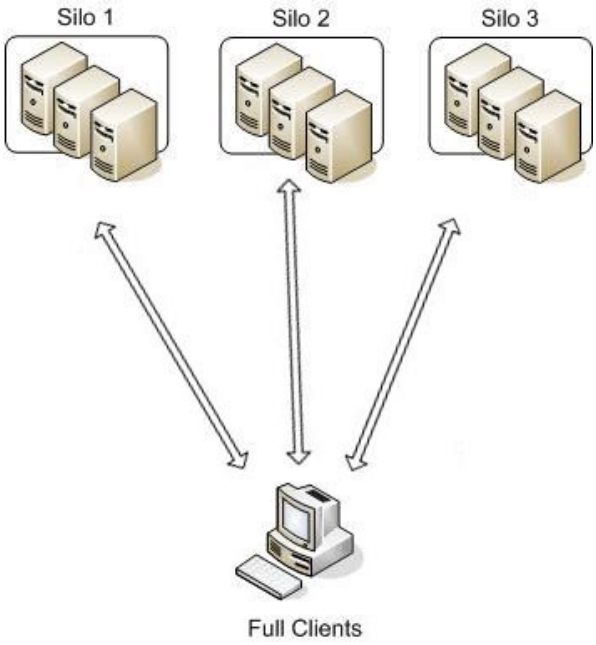
3.2.1 SBC-tekniikan toimintamalli

Käyttäjien ohjelmat asennetaan SBC farmin palvelimille, josta ohjelmat julkaistaan palveluina. SBC-järjestelmä erottelee näytölle ilmestyvän kuvan ohjelmien muista toiminnoista. Ohjelmat toimivat siis fyysisesti palvelimella, mutta ohjelmien käyttöliittymän tapahtumat näytetään käyttäjän työasemalla, erillisen asiakasohjelman kautta (Kuva 3.2). (Brown 2005: 8.)



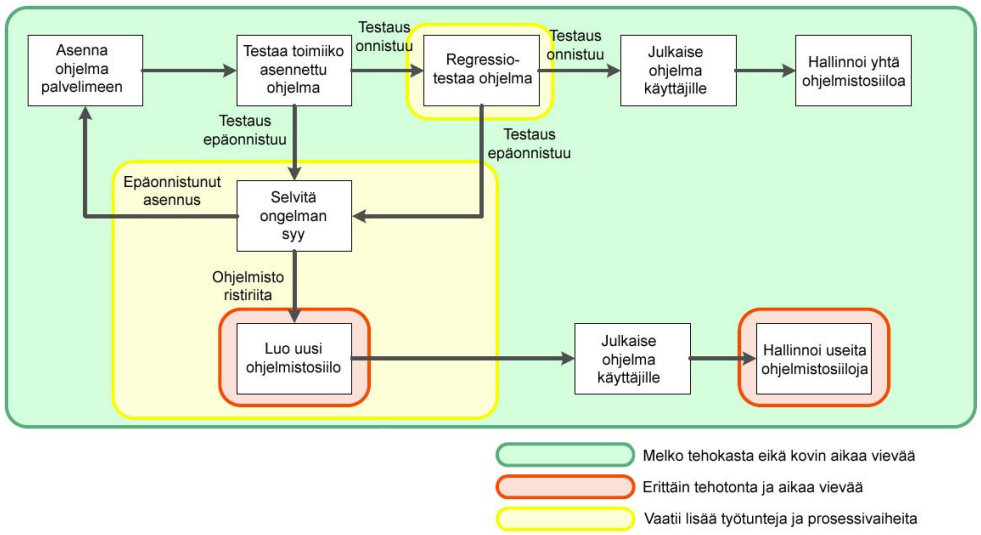
Kuva 3.2 SBC-tekniikka on tapa ”*esittää*” ohjelmia käyttäjille.

Ohjelmien asennusmenetelmät eivät eroa normaalista käytännöstä SBC-tekniikassa. Kaikki ohjelmat on fyysisesti asennettava farmin palvelimiin. Tästä johtuen SBC-järjestelmässä on samat asennusongelmat kuin henkilökohtaisissa työasemissa, ongelmat ovat vain siirtyneet käyttäjien työasemilta palvelimille. Lisäksi tällä tekniikalla ei ole mahdollista julkaista käyttäjille ohjelmia, jotka vaativat erityislaitteistoa (Esim. AutoCad), sillä ohjelmat toimivat palvelimilla.



Kuva 3.3 Ohjelmistosiilot helpottavat ohjelmien ristiriitatilanteissa.

Ratkaisuksi asennusongelmiin Citrix tarjoaa tuotteessaan ohjelmien eristämistä omiksi *ohjelmistosiiloiksi*. Käytännössä tämä tarkoittaa toisilleen ristiriitoja aiheuttavien ohjelmien eristämistä erillisille palvelimille, jotka ovat kuitenkin osana samaa palvelinfarmia (Kuva 3.3). Erilliset palvelimet vaativat yritykseltä lisää investointeja sekä ylläpitoresursseja. Mikäli eristettäviä ohjelmia on vain muutama, saattaa uusien palvelinten käyttöaste jäädä vähäiseksi. (Madden 2004)



Kuva 3.4 SBC-tekniikan toimintakuva. Lähde: Brown 2005: 9.

Kun tutkitaan kuvaa 3.4 ajankäytön ja hallinnoinnin kannalta, löytyy SBC-tekniikan prosesseista paljon samaa kuin ESD-tekniikastakin. Mikäli kaikki onnistuu, toimii tämä prosessi hyvin. Jos näin ei käykään, törmätään samoihin ongelmiin kuin ESD-tekniikassakin. Lisäksi kuluu enemmän aikaa, rahaa ja resursseja ohjelmistosiilojen rakentamiseen.

3.2.2 Ohjelmistojen hallintaprosessi SBC-tekniikalla

Ohjelmien asennus, ylläpito, päivitys ja poisto SBC-tekniikalla (Brown 2005: 8):

- **Asennus** – Käyttäjän näkökulmasta SBC-tekniikalla esitetään ohjelma hänen omalla koneellaan, joka vaikuttaa aivan kuin ohjelma olisi asennettu hänen koneelleen. Ohjelman näkökulmasta palvelin on kuin mikä tahansa työasema, johon se on fyysisesti asennettava. Toisin sanoen vaihtoehtoina ohjelmien asennuksille on edelleen vain manuaalinen sekä ESD-tekniikka, asennuskohde on vain muuttunut. Vaikka ohjelmien jakelu tai esittäminen on helpottunut, samat ongelmat ovat edelleen olemassa.

- **Ylläpito, päivitys ja poisto** – Koska ohjelmat on fyysisesti asennettava palvelimelle, koskee niiden ylläpitoa, päivitystä sekä poistoa samat toimenpiteet kuin manuaalista ylläpitoakin. Lisänä manuaaliseen ylläpitoon tulee ylläpitotoimenpiteiden ajankohdan suunnittelu ja käyttäjien informointi näistä toimenpiteistä, sillä manuaalisessa ylläpidossa vain yksi ihminen kerrallaan on käyttämättä ohjelmia, SBC-tekniikassa heitä voi olla paljon.

3.2.3 Citrix Presentation Server 4.0 (AIE)

Citrix Presentation Server 4.0 (with Application Isolation Environment) (AIE) on Citrix-yrityksen SBC-järjestelmän uusin versio, joka tuo mukanaan kaivatun uuden ominaisuuden, joka on ohjelmien eristäminen. Tätä osaa Citrix nimittää ohjelmien virtualisoinniksi. (Brown 2005: 10.)

AIE-komponentin tarjoamat edut ovat(Brown 2005: 10.):

- mahdollisuus asentaa ohjelmia, joita aikaisemmissa SBC-tekniikan palvelimiin ei ollut mahdollista asentaa
- mahdollisuus asentaa ja käyttää ohjelmia, jotka käyttävät eri versioita jaetuista komponenteista
- mahdollisuus päivittää ohjelmia ilman pelkoa, että jokin toinen ohjelma lopettaisi toimintansa
- mahdollisuus asentaa ohjelmia, joita ei ole suunniteltu asennettavan useamman käyttäjän ympäristöön

Tämä on saavutettu uudelleen ohjaamalla tiedostojärjestelmä ja rekisteri käyttäjä- tai ohjelmakohtaisesti AIE:n avulla.

Tiedostojärjestelmän uudelleen ohjaus – Käyttäjän ja ohjelman tiedostot ja hakemistopolut voidaan uudelleen ohjata toiseen hakemistopolkuun joko kokonaisuudessaan tai vain halutuin osin. Tämä tehdään käyttäjäkohtaisesti. Ylläpitäjä voi erillisten sääntöjen avulla määritellä mitkä tiedostot tai hakemistot ovat tai eivät ole uudelleen ohjattuja. Jokaisella säännöllä on turvamääritys, joka sallii tai kieltää lukemisen, kirjoittamisen ja suorittamisen. (Brown 2005: 11.)

Rekisterin uudelleen ohjaus – Kuten tiedostojärjestelmässäkin käyttäjän ja ohjelman rekisterimerkinnät voidaan uudelleen ohjata toiseen paikkaan rekisterissä AIE:n toimesta. Ylläpitäjä voi erillisillä säännöillä määritellä, mitkä merkinnät ovat tai eivät ole uudelleen ohjattuja. Jokaisella säännöllä on myös turvamääritys, joka sallii tai kieltää lukemisen, kirjoittamisen ja suorittamisen. (Brown 2005: 11.)

AIE-tekniikkaa käytettäessä pätee ohjelmien asennusprosessissa samat perusvaiheet kuin SBC-tekniikassakin. Lisäksi tarvitaan seuraavat vaiheet (Brown 2005: 11–12):

1. AIE tuki täytyy ottaa käyttöön palvelin farmille.
2. AIE:n polkuasetukset täytyy luoda uudelleen ohjausta varten.
3. Ohjelma täytyy asentaa AIE-ympäristöön. Tämä tehdään komentokehotetyökalulla AIESETUP, joka edellyttää ylläpitäjältä hyvää kyseisen ohjelman asennuksen ja toiminnan tuntemusta.

AIE-ympäristöä käytettäessä ohjelma on edelleen asennettava fyysisesti farmin palvelimelle. Muutoksena aikaisempaan on, että tiedostot ja rekiste-

rimerkinnät ohjataan eri paikaan palvelimella. Kun uudelleen ohjatusta tiedostosta tulee pyyntö, osoittaa AIE-ympäristö pyynnölle tiedoston tai rekisterimerkinnän uuden paikan.

Ohjelmien päivitys ja poisto vaativat myös normaalit toimenpiteet aivan kuin ennenkin.

3.2.4 SBC-tekniikan hyvät ja huonot puolet

Hyvät puolet:

- Ohjelmien fyysisten asennusten lukumäärä laskee SBC-tekniikalla huomattavasti ja AIE:n myötä vielä enemmän, koska ohjelmistosiiloja ei enää tarvita niin usein.

- Ohjelmien esittämisen hallinta on keskitetty, eli SBC-tekniikalla voidaan hallita mitä ohjelmia esitetään ja kenelle.

- SBC-tekniikka tarjoaa hyvän tietoturvan yrityksen kriittisimmille ohjelmille, koska ohjelmia käytetään yrityksen sisäverkossa ja ainoastaan ohjelmien tapahtumat näytetään käyttäjän työasemalla. Näin itse työasemille ei jää mitään dataa.

Huonot puolet:

- Ohjelmia asennettaessa on palvelin aina pois käytöstä. Tämä aiheuttaa käyttökatkoksia palvelun käyttäjille.

- Jos palvelin tai ohjelmat eivät toimi oikein, häiritsee tämä kaikkia palvelun käyttäjiä.

- Koska SBC-tekniikassa ohjelmat asennetaan fyysisesti palvelimeen, ovat ohjelmien ristiriidat yhtä mahdollisia kuin ESD-tekniikassakin. Regressio-testausta tarvitaan, jotta saadaan selville mitkä ohjelmat aiheuttavat ristiriitoja. Tämän jälkeen voidaan ohjelmat ohjata uudelleen eri paikkaan AIE:lla.

- Uudelleenohjaus on ns. ”jälkiviisautta”, sillä ylläpitäjän pitää joka tapauksessa ensin selvittää, mitkä ohjelmat eivät toimi keskenään.

- Kaikki ohjelmat eivät toimi terminaalipalvelimissa, esim. näytönohjaimesta tai prosessorista riippuvat ohjelmat.

- AIE vaatii vankkaa tietämystä mitkä ohjelman tiedostot, rekisterimerkinnät tulisi uudelleen ohjata.

- AIE ei toimi kaikissa ohjelmissa, esimerkiksi ohjelmat jotka vaativat uudelleen käynnistyksen asennuksessa.

3.3 Virtualisointitekniikka

Virtualisointi on terminä suomenkielessä yleisesti tuntematon. Termi on lähtöisin englanninkielen sanasta ”*virtualization*”, jolla Wikipedian mukaan tarkoitetaan seuraavaa:

- Virtualisoinnilla mahdollistetaan tietokoneiden laitteiden sekä muiden resurssien esittäminen käyttäjälle tai ohjelmalle niin, että resursseista saadaan helpommin kaikki niiden tarjoamat hyödyt irti.
- Virtualisointi tarjoaa paremminkin loogisen kuin fyysisen näkymän datasta, prosessointitehosta, levytilasta ja muista resursseista.

Virtualisointi termi on viimeaikoina ollut paljon esillä myös suomalaisissa tietotekniikan alan julkaisuissa.

3.3.1 Syitä virtualisoinnin syntymiseen

Nykypäivän tietojärjestelmien osat ovat monesti hyvin riippuvaisia toisistaan. Käyttöjärjestelmä on optimoitu tietyn tyyppiselle laitteistolle. Esimerkiksi Windows on suunnattu x86- ja x86-64 -arkkitehtuurin prosessoreille. Ohjelmat on kirjoitettu tietylle käyttöjärjestelmälle ja tietokannat on suunniteltu tietylle ohjelmalle. Näin tiiviissä kokonaisuudessa yhden osan muuttuminen ei voi olla vaikuttamatta ympäristöönsä. Nämä riippuvuudet muodostavat useimmin tietojärjestelmien ongelmat kuten:

- **Skaalattavuus** – Esimerkiksi useampia versioita Java Runtime Environment ohjelmatulkista ei ole voitu asentaa tai käyttää yhtä käyttöjärjestelmää kohti. Tämä rajoite rajaa pois käytöstä kaikki ohjelmat, jotka vaativat toimiakseen eri version kuin kyseiseen työasemaan on asennettuna.

- **Luotettavuus** – Esimerkiksi ei ole voitu olla varmoja aiheuttaako jonkin ohjelman asennus ongelmia tai ristiriitoja toisille ohjelmille.

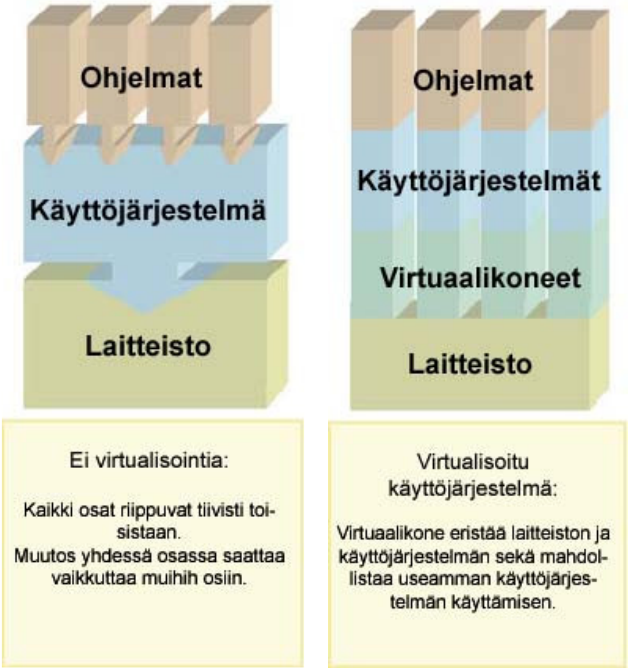
- **Hallittavuus** – Esimerkiksi miten voidaan hallita ohjelmien elinkaari työasemilla, jos ei voida olla varmoja, että muutokset eivät vaikuta muihin ohjelmiin tai käyttöjärjestelmään.

Virtualisoinnilla pyritään poistamaan näitä riippuvaisuuksia, tekemällä eri osista itsenäisiä ja ympäristöstään riippumattomia. (Wilson 2005: 2-3.)

Viime vuosina virtualisoituja osa-alueita on esitelty useita, kuten verkkoja (*VPN – Virtual Private Network*), tallennuskapasiteettia (*Storage*) sekä tietokoneita ja käyttöjärjestelmiä. Viimeisestä sekä tietokoneen että käyttöjärjestelmän virtualisointi tekee molemmista riippumattomia toisistaan. Näin tietokoneen fyysistä prosessointitehoa voidaan hyödyntää useille eri työkuormille, kuten useille eri käyttöjärjestelmille ja niiden eri ohjelmille. Virtualisoinnissa on siis pyrkimys poistaa riippuvaisuutta fyysisten resurssien väliltä ja tehdä niistä loogisia.(Wilson 2005: 3.)

Esimerkki virtualisoinnin tuomista eduista: Yrityksellä on oltava tarpeeksi palvelinkapasiteettia, jotta lyhytkestoisimmistakin piikkikuormista selvittäisiin. Ilman virtualisointia palvelimeen ei välttämättä voida lisätä muita käyttöjärjestelmiä tai ohjelmia, jolloin sen keskimääräinen käyttöaste saattaa jäädä 10 – 15 %. (Wilson 2005: 3.)

Eriyttämällä käyttöjärjestelmä fyysisestä laitteistosta virtualisoinnilla, saadaan yhden palvelimen käyttöastetta nostettua huomattavasti (Kuva 3.5). Näin palvelimeen voidaan asentaa useampia käyttöjärjestelmiä ja samalla fyysisten palvelinlaitteistojen määrää saadaan laskettua. (Wilson 2005: 3.)



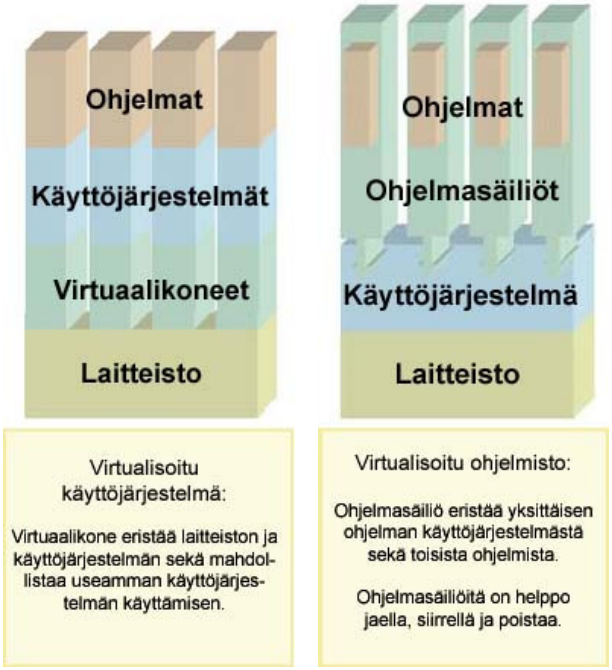
Kuva 3.5 Virtualisoinnin tuoma helpotus palvelimiin ja työasemiin. Lähde: Wilson 2005: 4.

Käyttöjärjestelmän virtualisoinnista huolimatta, ohjelmat ovat vielä riippuvaisia käyttöjärjestelmästään. Ohjelmien asennusongelmista johtuen yksittäinen ohjelma saattaa vaatia oman käyttöjärjestelmän toimiakseen. Tämänkaltaiset ”yksi ohjelma per yksi käyttöjärjestelmä” -paketit varaavat turhaan laitteiston fyysisiä resursseja lisäämällä tarvittavien virtuaalikoneiden määrää sekä vähentämällä mahdollisten ohjelmien määrää. (Wilson 2005: 3.)

Esimerkiksi, jos yksi ohjelma vaatisi 10 % tietokoneen prosessointitehosta ja yksi käyttöjärjestelmä vaatisi myös 10 % tietokoneen prosessointitehosta, voisi tietokone yhdellä käyttöjärjestelmällä pitää yllä yhdeksää samankaltaista ohjelmaa. Mikäli taas ohjelma aiheuttaa ristiriitoja ja vaatisi toimiakseen aina oman käyttöjärjestelmän, kuluttaisi käyttöjärjestelmän ja ohjelman yhdistelmä yhteensä 20 % prosessointitehoa. Näitä yhdistelmiä voisi tietokoneessa olla enää vain viisi. (Wilson 2005: 3.)

3.3.2 Ohjelmistojen virtualisointitekniikka

Ohjelmistojen virtualisointitekniikalla on pyritty ratkaisemaan käyttöjärjestelmän ja ohjelmien välisiä riippuvuuksia samalla tavalla kuin tietokoneen ja käyttöjärjestelmän tapauksessa – eristämällä nämä kaksi käsitettä toisistaan (Kuva 3.6). (Wilson 2005: 4.)



Kuva 3.6 Miten ohjelmisto- ja käyttöjärjestelmävirtualisointi eroavat toisistaan. Lähde: Wilson 2005: 4.

Eristäminen on toteutettu ns. loogisella ”*säiliöllä*”. Säiliö pitää sisällään oman tiedostojärjestelmän sekä kaikki ohjelman käynnistämiseen ja käyttämiseen tarvittavat tiedostot. Säiliö sisältää myös oman rekisterin, jossa on kaikki ohjelmalle tarpeelliset merkinnät. Säiliön ulkopuolelle jää itse käyttöjärjestelmä sekä käyttäjäkohtaiset asetukset ja tiedostot. (Wilson 2005: 4.)

Säiliöityjä ohjelmia ei asenneta, vaan ne siirretään tai jaellaan käyttäjien työasemille, joissa niitä tarvitaan. Säiliöidyt ohjelmat käynnistyvät aina paikallisesti säiliön sisällä. Paikallisuudesta johtuen ohjelma ei tarvitse jatku-

vaa verkkoyhteyttä ja ohjelma käyttää työaseman paikallisia resursseja, ku ten prosessoria ja muistia. (Wilson 2005: 4.)

Säiliöidyt ohjelmat eivät kirjoita mitään säiliön ulkopuolelle, kuten käyttöjärjestelmän rekisteriin, eivätkä ne kopioi dll-tiedostoja työasemalle, koska kaikki tarvittava löytyy säiliön sisältä. Lopputuloksena käyttöjärjestelmä pysyy muuttumattomana eli puhtaana. (Brown 2005: 13.)

Todellisen ohjelmien virtualisointitekniikan määritelmä on:

1. se muuttaa ohjelmien asennus- ja hallintatapoja
2. ohjelmat pysyvät täysin eristettyinä käyttöjärjestelmästä ja toisista ohjelmista
3. ohjelmat eivät kirjoita yhtään mitään käyttöjärjestelmään. Lopputuloksena käyttöjärjestelmä pysyy puhtaana
4. ohjelmia ei missään muodossa asenneta käyttöjärjestelmään

Todellisen ohjelmien virtualisointitekniikan tarkoituksena on:

1. tehdä henkilökohtaisesta työasemasta yleinen tietokone, joka mukautuu eri käyttäjille määriteltyihin tarpeisiin
2. pystyä reaaliajassa tarjoamaan käyttäjälleen juuri ne ohjelmat joita hän tarvitsee ja joihin hänelle on määritetty käyttöoikeus

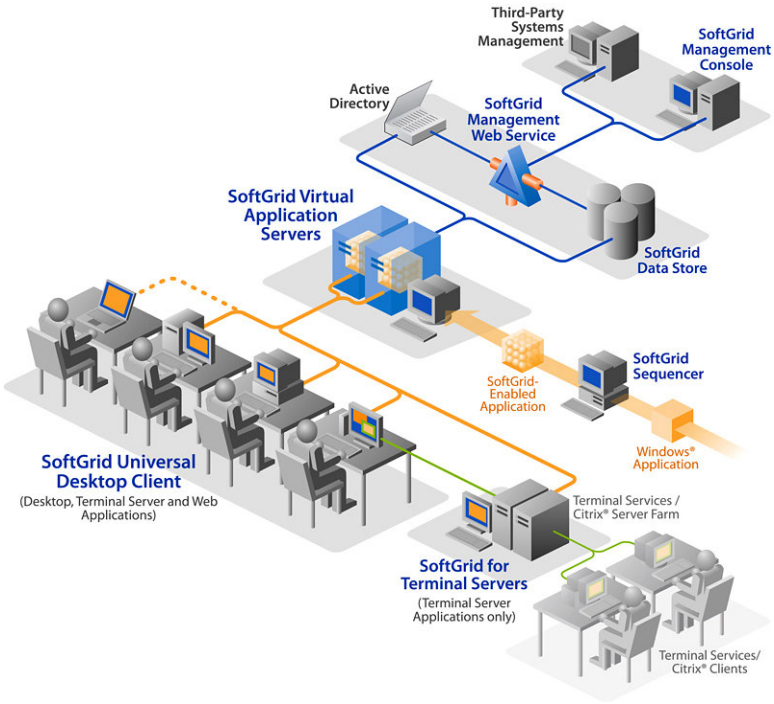
4 Ohjelmistojen ylläpito SoftGrid-tekniikalla

Softricity on yritys, joka on perustettu vuonna 1999. Alkuperäinen idea lähti liikkeelle vuonna 1996, kun David Greschler, Stuart Schaefer ja Owen Mysliwy kehittivät uuden tavan jakaa ohjelmia Bostonin tietokonemuseon näyttelyyn. He pystyivät nopeasti jakamaan 15 tietokoneelle käyttöön yli 40 ohjelmaa. (Softricity Company 2005)

SoftGrid on Softricityn markkinoille tuoma järjestelmä, jonka tarkoituksena on parantaa yritysten Windows-pohjaisten tietokoneiden, kuten henkilökohtaisien työasemien, kannettavien ja terminaalipalvelimien (SBC) ohjelmien asennusta sekä hallintaa. (Softricity Products 2005)

4.1 SoftGrid-järjestelmä

Kuvan 4.1 SoftGrid-järjestelmä mahdollistaa käyttöjärjestelmästä virtuaalisesti eristettyjen ohjelmien eli ohjelmasäiliöiden rakentamisen sekä niiden keskitetyn jakelun ja hallinnan.



Kuva 4.1: SoftGrid-järjestelmä. Lähde: http://www.softricity.com

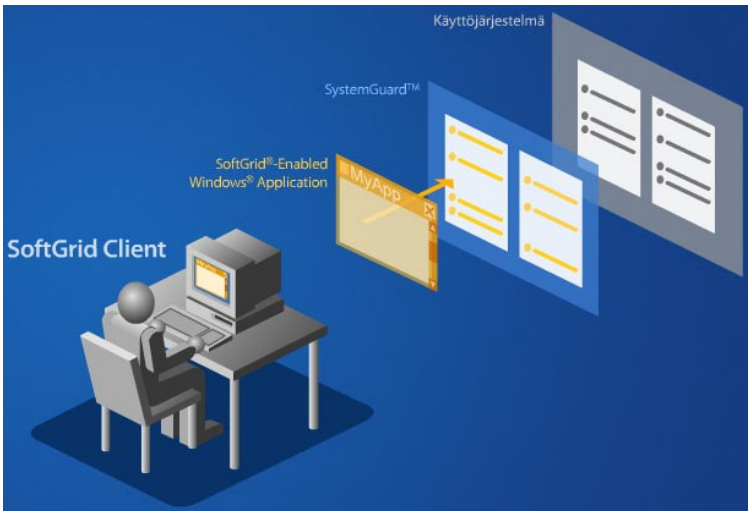
SoftGrid-järjestelmä koostuu useasta eri komponentista (Softricity Desktop 2005):

- hallintakomponenteista kuten *SoftGrid Management Web Service*, *SoftGrid Management Console* ja *SoftGrid Data Store*
- *SoftGrid Virtual Application Server* -palvelimesta
- työasemille asennettavasta *SoftGrid Universal Desktop Client* -asiakasohjelmasta
- terminaalipalvelimille asennettavasta *SoftGrid Client for Terminal Servers* -asiakasohjelmasta
- *SoftGrid Sequencer* ohjelmistojen paketointiohjelmasta

SoftGrid-järjestelmän komponentit käydään tarkemmin läpi seuraavissa kappaleissa.

4.1.1 SoftGrid SystemGuard

Koko SoftGrid-järjestelmän sydämenä toimii Softricityn patentoima virtuaalinen ohjelmasäiliö, jota Softricity kutsuu nimellä SystemGuard. SystemGuard mahdollistaa paikallisten ohjelmien käytön erillään toisista ohjelmista ja alla olevasta käyttöjärjestelmästä. (Softricity T&C 2004: 46 – 47.)



Kuva 4.2: SystemGuard toimii rajapintana ohjelman ja käyttöjärjestelmän välissä. Lähde: http://www.softricity.com

Käsitteenä SystemGuard on ohjelmallinen eli looginen rajapinta käyttöjärjestelmän ja ohjelman välissä. SystemGuard on nimensä mukaisesti kuin vartija, joka hallitsee ohjelman toimintaa (Kuva 4.2).

Esimerkiksi, kun käyttäjä käynnistää SoftGrid-järjestelmällä työasemalle jaetun ohjelman, käynnistyy taustalle aina ennen ohjelmaa SystemGuard ja

vasta tämän jälkeen käynnistyy itse ohjelma. Ennen ohjelman käynnistystä SystemGuard avaa ohjelman tarvitsemat tiedostot, rekisterimerkinnät, ympäristömuuttujat ja fontit käyttöön omaan virtuaaliseen ohjelmasäiliöön. Mikäli ohjelman täytyy saada lukea esimerkiksi rekisterimerkintä, jota ohjelmasäiliössä ei ole, saa ohjelma luvan lukea merkinnän käyttöjärjestelmästä. Mutta jos ohjelma haluaa itse kirjoittaa rekisterimerkinnän, ohjaa SystemGuard tämän merkinnän ohjelmasäiliön omaan virtuaalirekisteriin. (Softricity T&C 2004: 46 – 47.)

SystemGuard virtualisoi ohjelmilta monta erilaista käyttöjärjestelmän komponenttia kuten (Softricity T&C 2004: 46 – 47):

- **COM-komponentit:** SystemGuard eristää COM-komponenttien keskustelun, suojaten näin ohjelmaa muilta ohjelmilta.

- **Rekisteri:** SystemGuard luo oman rekisterin kaikille sen kautta käytettäville ohjelmille. Nämä rekisterimerkinnät eivät ole näkyviä muille ohjelmille, mukaan lukien Windows Regedit ohjelma. Mikäli virtuaalirekisteristä ei löydy ohjelman pyytämää merkintää, on ohjelmalla mahdollisuus lukea merkintä käyttöjärjestelmän rekisteristä, mutta ohjelma ei silti voi kirjoittaa käyttöjärjestelmän rekisteriin. Ohjelman kirjoittamat merkinnät syntyvät virtuaalirekisteriin.

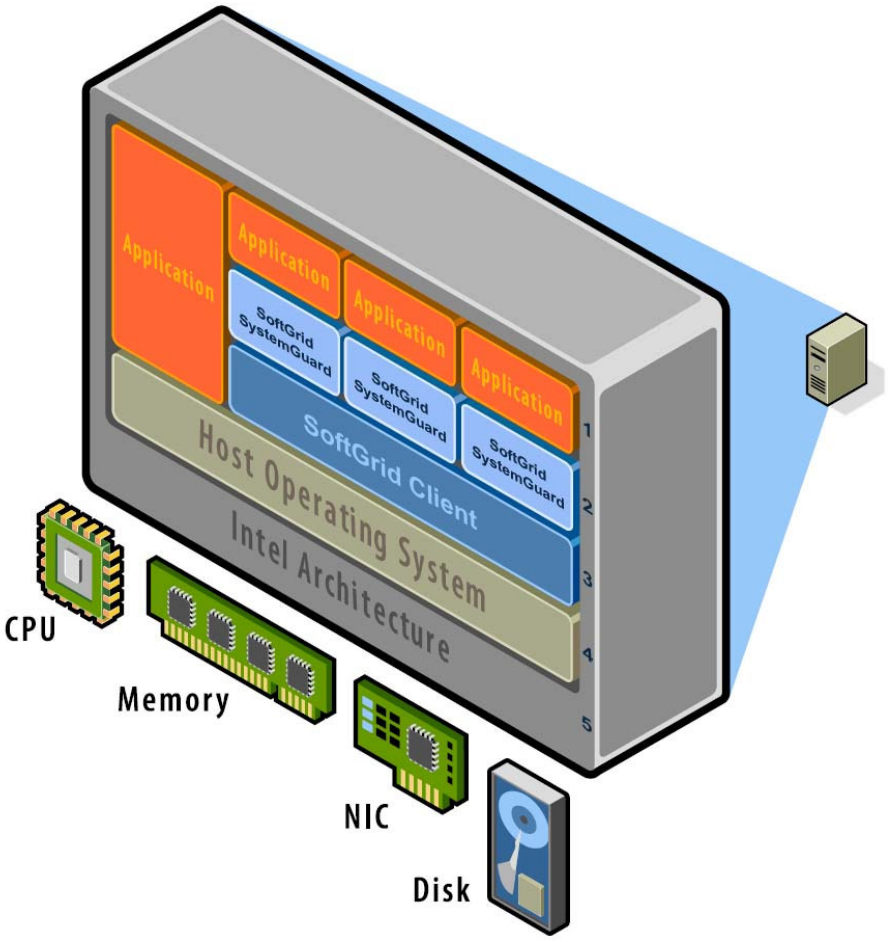
- **Tiedostojärjestelmä:** SystemGuard käsittelee myös ohjelman tekemät pyynnöt tiettyihin tiedostoihin. Esimerkiksi, jos ohjelma pyytää tiedostoa, joka sijaitsee paikallisella C: -asemalla, SystemGuard ohjaa tämän pyynnön sen omaan virtuaaliseen tiedostojärjestelmään. Tämä on erittäin tärkeä ominaisuus juuri dynaamisesti linkitettyjen kirjastojen osalta (.DLL), ettei ohjelma pääse käyttämään käyttöjärjestelmässä mahdollisesti paikallista versiota tai toisen ohjelman versiota kyseisestä DLL-tiedostosta.

- **.INI-tiedostot:** SystemGuard antaa jokaiselle ohjelmalle tai sen käytössä olevalle prosessille oman version .INI-tiedostoista muokattavaksi.

- **Ympäristömuuttujat:** SystemGuard sisältää omat ympäristömuuttujat.

- **Fontit:** SystemGuard sisältää myös ohjelman fontit, jotka ovat käytössä vain kyseiselle ohjelmalle sen käytön ajan.

Vaikka SystemGuard eristääkin ohjelmien näkyvyyden toisilleen sekä käyttöjärjestelmälle, ei ohjelmien pääsyä tarpeellisiin palveluihin kuten kopiointiin, leikkaamiseen tai liittämiseen estetä. Samoin ovat sallittuja toimenpiteitä esimerkiksi tulostus, töiden talletus koneen kovalevylle tai verkkolevylle.(Softricity T&C 2004: 46 – 47.)



Kuva 4.3: SoftGrid tietokoneen näkökulmasta.

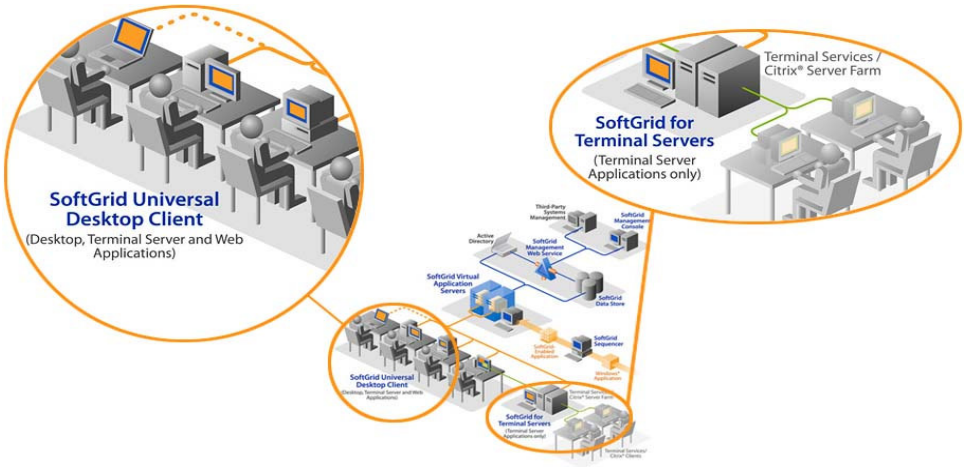
Työaseman näkökulmasta SoftGrid-asiakasohjelma, SystemGuard sekä SoftGrid-järjestelmällä jaettavat ohjelmat asettuvat tietokoneelle seuraavanlaiseen hierarkiaan (Kuva 4.3):

1. korkeimmalla tasolla sijaitsevat eristetyt ohjelma
2. jokaiselle ohjelmalle on oma SystemGuard rajapinta
3. kaikki ohjelmat käynnistyvät SoftGrid Client -asiakasohjelman kautta
4. Windows-käyttöjärjestelmä
5. alimmalla tasolla on laitteisto eli prosessori, muisti, verkkokortti, kovalevy, jne.

SystemGuard on myös osana molempia (*SoftGrid Universal Desktop Client* ja *SoftGrid Client for Terminal Servers)* asiakasohjelmia sekä (*SoftGrid Sequencer*) paketointiohjelmaa.

4.1.2 SoftGrid Client

Kuvan 4.4 SoftGrid-asiakasohjelmat (*SoftGrid Universal Desktop Client* tai *SoftGrid Client for Terminal Servers)* ovat vastuussa ohjelmien julkaisusta käyttäjän työasemalla ja yhteyksien muodostamisesta SoftGrid-palvelimelle. SoftGrid Client on myös vastuussa SystemGuard rajapinnan käynnistämisestä ohjelmille sekä ohjelmien lataamisesta ja niiden käynnistämisestä työasemalla. (Softricity T&C 2004: 54.)



Kuva 4.4: SoftGrid Client -asiakasohjelman sijoittuminen SoftGrid-järjestelmässä. Lähde: http://www.softricity.com

Käyttäjälle SoftGrid Client -asiakasohjelma näkyy lähinnä suorituspalkkina näytöllä ohjelmaa käynnistettäessä tai ohjauspaneelista löytyvänä ohjelmana, jolla voidaan vaihtaa SoftGrid Client -asiakasohjelman asetuksia.(Softricity T&C 2004: 54.)

SoftGrid Client -asiakasohjelma itsessään on hyvin kevyt ohjelma, joka vaatii tietokoneelta vähintään Windows 2000 käyttöjärjestelmän tai uudemman sekä samat resurssit, joita ohjelmat normaalistikin työasemalta vaativat toimiakseen. Asiakasohjelmien tarkat vaatimukset löytyvät liitteestä 1.

SoftGrid Client -asiakasohjelma toiminta:

1. Kun käyttäjä kirjautuu työasemalleen, SoftGrid Client -asiakasohjelma pyytää SoftGrid-palvelimelta listan käyttäjälle myönnetyistä ohjelmista ja julkaisee niistä pikakuvakkeen käyttäjän työpöydälle Pikakäynnistys-valikkoon, Käynnistä-valikkoon tai haluttaessa kaikkiin näihin paikkoihin.

2. Kun käyttäjä käynnistää SoftGrid-järjestelmällä luodun ohjelman, SoftGrid Client -asiakasohjelma lataa palvelimelta kyseisen ohjelman työasemalle asiakasohjelman välimuistiin, avaa SystemGuard-

säiliön taustalle ja käynnistää ohjelman SystemGuard-säiliössä. Mikäli ohjelma on aiemmin käynnistetty ja on jo ladattuna työasemalla asiakasohjelman välimuistissa, tarkastaa SoftGrid Client -asiakasohjelma palvelimelta ainoastaan, ettei ohjelmaan ole tullut päivityksiä ja että käyttäjällä on edelleen oikeus käyttää ohjelmaa.

3. Käyttäjän sulkiessa ohjelman, SoftGrid Client -asiakasohjelma hoitaa SystemGuard-säiliön sulkemisen ja käyttäjän mahdollisesti tekemien asetusten tallentamisen.

SoftGrid Client keskustelee SoftGrid-palvelimen kanssa käyttäen RTSP (Real Time Streaming Protocol, käyttäen porttia 554) tai RTSPS (Real Time Streaming Protocol Secure, käyttäen porttia 332) RFC-dokumentoituja protokollia. (Softricity T&C 2004: 54.)

RTSP-protokolla on alun perin suunniteltu ääni- ja videodatan kuljettamiseen, koska se mahdollistaa useampien yhtäaikaisten yhteyksien muodostamisen. Lähetettävä data pilkotaan eli *sekvensoidaan* ennen lähettämistä helposti kuljetettaviksi paloiksi. (RFC 2326)

RTSPS-protokolla on RTSP-protokollan jalostettu ja turvallisempi versio, joka noudattaa käytännössä samaa salaustekniikkaa, kuin HTTPS-protokolla eli käytettävä salausprotokolla on TLS 1.0 (*Transport Layer Security*) (RFC 2246), joka perustuu SSL 3.0 protokollaan (Secure Socket Layer).

SoftGrid-järjestelmä hyödyntää näitä protokollia jaettavien ohjelmasäiliöiden lähetyksessä. Ohjelmasäiliöitä luotaessa SoftGrid Sequencer -paketointiohjelma sekvensoi ohjelman tiedostot jo valmiiksi lähetystä varten. Tämä helpottaa SoftGrid-palvelimelta isojen ohjelmasäiliöiden lähettämistä samanaikaisesti SoftGrid Client -asiakasohjelmille. Lisäksi SoftGrid-järjestelmä pystyy RTSP- tai RTSPS-protokollaa hyödyntäen toimittamaan ohjelmasäiliöistä vain sen osuuden, jota ohjelman käynnistämiseen aluksi tarvitaan. Kun ohjelma käyttää osaa, jota ei palvelimelta ole vielä saatu, pyytää SoftGrid Client -asiakasohjelma palvelimelta lisää ohjelman osia. Ylläpitäjä voi määritellä SoftGrid Client -asiakasohjelman myös hakemaan koko ohjelmasäiliön.

SoftGrid Client -asiakasohjelma säilyttää lataamiansa ohjelmasäiliöitä välimuistissa työaseman kovalevyllä, josta ohjelmasäiliöt ovat käytettävissä vaikka työasemalla ei aina lähiverkkoon olisikaan yhteyttä. Yhteys lähiverkkoon on kuitenkin välttämätön ohjelmasäiliön ensimmäisellä käynnistyskerralla, sillä samalla tarkastetaan käyttäjän oikeus käyttää kyseistä ohjelmaa. Lähiverkon yhteys on myös välttämätön, mikäli käyttäjällä ei ole koko ohjelmasäiliötä työasemallaan ja hän käyttää ohjelmasta osaa, jota työasemalle ole vielä ladattu.

Ohjelmasäiliöt toimivat välimuistista oletuksena 65535 minuuttia (45 päivää) ja tämä arvo voi olla väliltä 1 – 65535 minuuttia (Softricity T&C 2004: 96). Myös välimuistin kokoa kovalevyllä voidaan säätää. Oletuksena välimuistin koko on SoftGrid Universal Desktop Client -asiakasohjelmalla 2048 MB ja SoftGrid Client for Terminal Servers -asiakasohjelmalle 4096 MB. Molempien asiakasohjelmien välimuistia voidaan kasvattaa 64 GB asti.(Softricity T&C 2004: 100.)

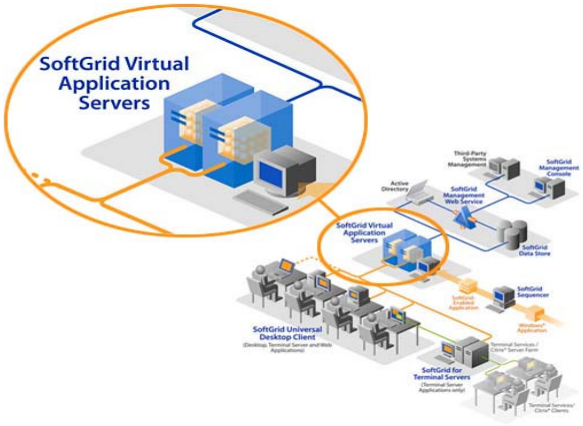
Koska SoftGrid Client ei vaadi jatkuvaa yhteyttä palvelimeen, pystytään sillä palvelemaan yhdellä palvelimella huomattavasti useampaa käyttäjää kuin esimerkiksi SBC-järjestelmällä. RTPS-protokolla ja sekvensointi helpottavat myös osaltaan ohjelmien latausta vaikka lähiverkon nopeus ei olisikaan suurempi kuin 10Mbs.

SoftGrid Client -asiakasohjelmien välillä ei ole muuta eroa kuin että SoftGrid Universal Desktop Client on tarkoitettu työasemille sekä palvelimille ja SoftGrid Client for Terminal Servers on tarkoitettu SBC-järjestelmille joko Windows- tai Citrix-järjestelmälle.

SBC-järjestelmälle SoftGrid Client -asiakasohjelman toiminta on täysin samanlaista kuin työasemaversiossakin. Asiakasohjelma kuitenkin mahdollistaa uusien ohjelmien käyttöönoton ilman katkoksia ja regressio-testausta Terminal -palvelimilla.

4.1.3 SoftGrid Virtual Application Server

SoftGrid Virtual Application Server eli SoftGrid-palvelin valvoo ohjelmien käyttöä ja tallentaa käyttöä koskevat tiedot SoftGrid-tietokantaan (Kuva 4.5). Se varastoi virtualisoituja ohjelmia eli ohjelmasäiliöitä ja jakaa niitä SoftGrid Clientille sen pyynnöstä. Tästä syystä SoftGrid-palvelimen rooli on tärkeä ja sen 100 % vikasietoisuus tulisi varmistaa esimerkiksi useammalla SoftGrid-palvelimella ja verkkokuorman jakotekniikalla (NLB – Network Load Balancing). (Softricity T&C 2004: 53.)



Kuva 4.5: SoftGrid -palvelimen sijoittuminen SoftGrid -järjestelmään. Lähde: http://www.softricity.com

SoftGrid-palvelin vaatii toimiakseen Windows Server 2000 tai Windows Server 2003 -tuoteperheen käyttöjärjestelmän, SoftGrid-hallintakomponentit sekä Microsoft Active Directory tai NT 4.0 toimialueen. Tarkemmat vaatimukset löytyvät liitteestä 1.

SoftGrid -palvelimen toiminta (Kuva 4.6):

1. Kun SoftGrid Client -asiakasohjelmalta saapuu pyyntö ohjelmista tietylle käyttäjälle, kysyy SoftGrid-palvelin ensin SoftGrid-tietokannasta:
   - Kenelle on annettu oikeus käyttää ohjelmia?
   - Onko ohjelmalle määritetty lisenssiä ja onko niitä vapaana?

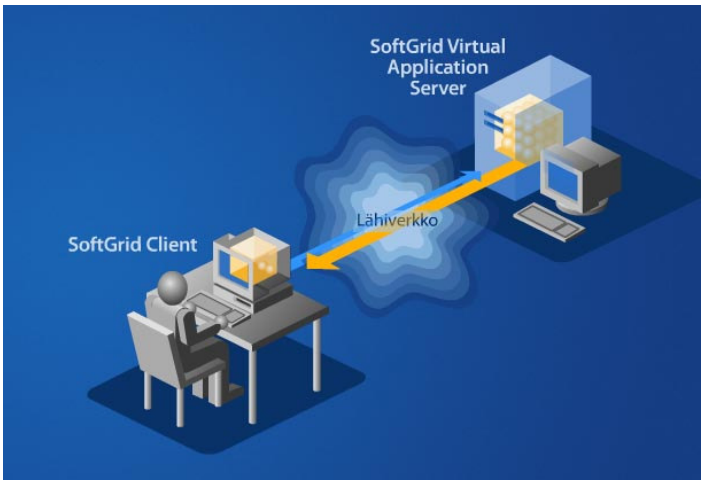
2. Tämän jälkeen SoftGrid-palvelin kysyy Microsoft toimialueelta (Active Directory tai Windows NT 4.0 domain) onko kyseinen käyttäjä olemassa ja kuuluuko hän kyseisiin ryhmiin, jotka SoftGrid-tietokanta ohjelmille ilmoitti.

3. Kun käyttäjän oikeudet ohjelmiin on tunnistettu, lähettää SoftGrid-palvelin tiedot ohjelmista SoftGrid Client -asiakasohjelmalle.

4. Käyttäjän käynnistäessä SoftGrid-järjestelmällä jaetun ohjelman työasemallaan, pyytää SoftGrid Client -asiakasohjelma käynnistettyä ohjelmasäiliötä SoftGrid-palvelimelta.

5. SoftGrid-palvelin lähtee jakamaan ohjelmasäiliötä SoftGrid Client -asiakkaalle pieninä paloina.

6. Aina kun ohjelmasta käytetään sellaista osaa, jota ei palvelimelta ole vielä haettu, pyytää SoftGrid Client -asiakasohjelma lisää ohjelman palasia SoftGrid-palvelimelta.

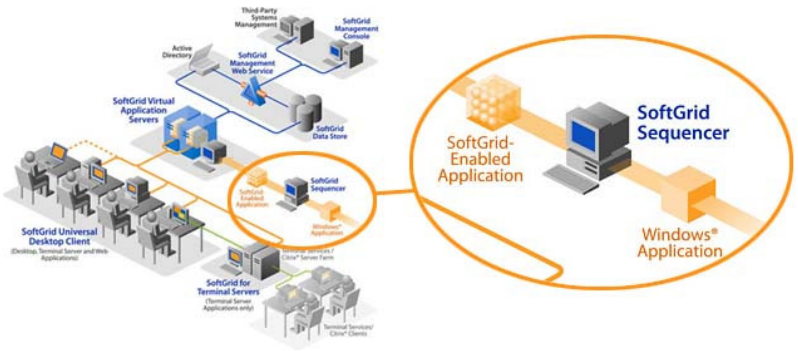


Kuva 4.6: SoftGrid Client -asiakasohjelman ja SoftGrid-palvelimen välinen toiminta. Lähde: http://www.softricity.com

Mikäli käyttäjälle ei ole myönnetty oikeutta mihinkään ohjelmaan, ei hänelle myöskään jaella yhtään ohjelmaa.

4.1.4 SoftGrid Sequencer

SoftGrid Sequencer on SoftGrid-järjestelmän osa, jolla luodaan asennettavista ohjelmista valmiiksi asennettuja, itsenäisiä ohjelmasäiliöitä (Kuva 4.7).



Kuva 4.7: SoftGrid Sequencerin sijoittuminen SoftGrid -järjestelmään. Lähde: http://www.softricity.com

Sequencer on velhopohjainen (Wizard-based) ohjelma, joka asennetaan erilliseen työasemaan. Kun ohjelmia paketoidaan, paketointityöasemassa ei saa olla mitään muuta asennettuna kuin käyttöjärjestelmä ja Sequencer-paketointiohjelma, jottei asennusongelmia pääse syntymään ohjelmasäiliötä luotaessa. (Softricity T&C 2004: 201.)

Samoin kuin ESD-tekniikan paketointiohjelmat, Sequencer-paketointiohjelma tarkkailee paketoitavan ohjelma asennusta. Asennuksen jälkeen Sequencer luo asennuksessa syntyneistä muutoksista ja tiedostoista erillisen ohjelmasäiliön. (Softricity T&C 2004: 201.)

Toisin kuin ESD-tekniikan paketointiohjelmat, Sequencer-paketointiohjelma lisää ohjelmasäiliöön uusia ominaisuuksia, joiden avulla ohjelmaa ei tarvitse enää erikseen asentaa jokaiselle työasemalle. Ennen ohjelman asennuksen käynnistämistä, Sequencer käynnistää SystemGuard-säiliön. Asennuksessa tapahtuneiden muutosten perusteella Sequencer pystyy luomaan valmiit asetukset ohjelmasäiliölle. Lisäksi Sequencer sekvensoi eli jakaa asennetun ohjelman luomat tiedostot osiin ohjelmasäiliön sisään, niin että ohjelma voidaan jaella pienissä osissa työasemille. Tämä mahdollistaa ohjelman jakelun vain niiltä osin kuin se on tarpeellista. Esimerkiksi Microsoft Wordista ei tarvitse jaella kaikkia ominaisuuksia käyttäjälle, mikäli hän ei niitä tarvitse. (Softricity T&C 2004: 201.)

Ohjelmasäiliön luonnin aikana Sequencer pyytää myös käynnistämään ohjelman samalla tarkkaillen mitä tiedostoja ja asetuksia ohjelma tarvitsee käyttöjärjestelmän puolelta. Tällä varmistetaan, että ohjelmasäiliö sisältää myös mahdolliset käyttöjärjestelmältä vaadittavat tiedostot ja rekisterimerkinnät. Tämä on tärkeää, koska näin ohjelmasäiliötä on mahdollista jakaa työasemille käyttöjärjestelmän versiosta riippumattomana. Esimerkiksi Windows 2000, XP ja 2003 käyttöjärjestelmille. Rajoite tähän tulee lähinnä ohjelman puolelta, mikäli sitä ei ole suunniteltu tukemaan kuin tiettyjä Windows-versioita.

Ennen ohjelmasäiliön rakentamista tulisi kuitenkin miettiä miten ohjelmat kommunikoivat toisilleen. Yhtenä SoftGrid-järjestelmän pääkomponenttina toimii SystemGuard, jonka tarkoituksena on eristää ohjelmia ja estää niitä kommunikoimasta toisilleen. Tämä tietenkin estää toiminnan myös ohjelmilta, joiden normaalisti tarvitsisi kommunikoida toisilleen (esimerkiksi Microsoft Office -ohjelmat ovat useimmissa tapauksissa hyvin riippuvaisia toisistaan). Excel ja Access ohjelmasta voidaan luoda datalinkkejä toisiinsa tai muihin Office-ohjelmiin, joiden tarkoituksena on päivittää tietoja automaattisesti toisiin ohjelmiin, kun data niissä itsessään muuttuu. Microsoft Office-ohjelmat ovat vain yksi esimerkki tämän tapaisista ohjelmistoista. Ohjelmien syvästi toisiinsa integrointi ei välttämättä ole pelkästään valmistajakohtaista, vaan myös eri valmistajien ohjelmat saattavat tarvita toisiaan toimiakseen oikein.

Tämän ongelman ratkaisemiseen SoftGrid-järjestelmällä on muutamia eri vaihtoehtoja. Oleellisinta on ottaa selville miten ja minne virtualisoitava ohjelma kommunikoi. Virtuaaliympäristöstä ohjelman annetaan kommunikoida seuraavilla eri tavoilla (Softricity T&C 2004: 200):

Tiedostoassosiaatiot = tiedostopäätteiden ja ohjelmien välinen suhde. Jos esimerkiksi virtuaaliympäristö saa ohjelman käynnistyspyynnön, joka ei ole

samassa virtuaaliympäristössä, ohjaa käyttöjärjestelmä tämän pyynnön sille ohjelmalle, joka kyseistä tiedostopäätettä on määritetty käyttämään, vaikka kyse olisikin virtualisoidusta ohjelmasta.

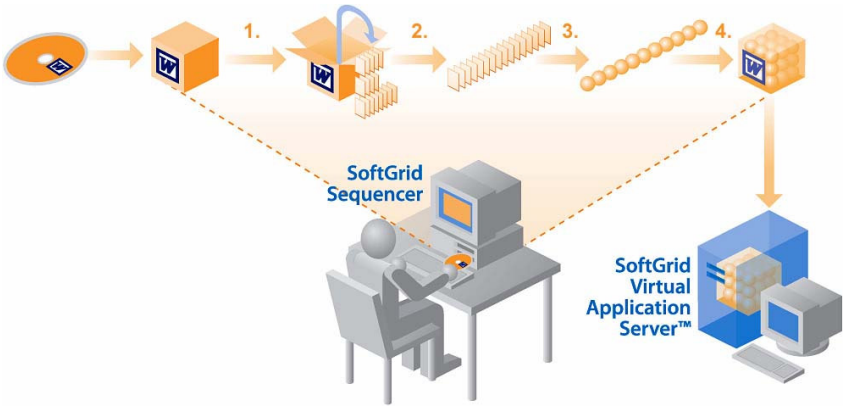
Leikepöytä = Kopioi, leikkaa ja liitä toiminnot ovat sallittuja ohjelmien käyttäjille virtuaaliympäristöjen välillä.

Paikalliset ohjelmat = pysyvästi ja perinteisin menetelmin käyttöjärjestelmään asennetut ohjelmat. Jos ohjelma ei löydä etsimäänsä tietoa virtuaaliympäristöstä, jatkaa ohjelma tiedon etsimistä virtuaaliympäristön ulkopuolelta, joko kysyen asiasta käyttöjärjestelmästä tai paikallisesti asennetulta ohjelmalta.

Mikäli kaksi tai useampi ohjelma kommunikoivat toisilleen jollain toisella tavalla kuin yllämainituilla tavoilla, tulisi niistä luoda yhteinen ohjelmistosäiliö eli asentaa ohjelmat yhtä aikaa Sequencer-ohjelmassa yhteiseen virtuaaliympäristöön. Tätä tapaa kutsutaan SoftGrid-järjestelmässä nimellä **Suite** eli *ohjelmistokokoelma*. Softricity suosittelee asentamaan aina ohjelmat, joita ei ole suunniteltu käyttämään toisiaan hyväksi tai joiden ei tarvitse keskustella toistensa kanssa, erillisiin ohjelmasäiliöihin ristiriitojen välttämiseksi. Kaikki Microsoft Office -ohjelmat ovat kuitenkin niin integroituja toisiinsa, että ne tulisi poikkeuksetta asentaa yhteiseen virtuaaliympäristöön.(Softricity T&C 2004: 202.)

Sequencer-paketointiohjelma luo asennetusta ohjelmasta neljä tiedostoa (Softricity T&C 2004: 202):

- **SFT** – Tiedosto sisältää kaikki asennetun ohjelman tiedostot paketoituna yhdeksi tiedostoksi.
- **OSD** (Open Software Description) – XML-kuvaustiedostoa käytetään antamaan informaatiota SoftGrid Client -asiakasohjelmalle virtuaalisen ohjelman käynnistämisestä. Kyseinen tiedosto voidaan mieltää normaalisti asennetun ohjelman käynnistystiedostoksi. Tiedosto sisältää muutakin tärkeää tietoa kuten vaatimukset ohjelman käynnistämiseen, ohjelman latauspalvelimen tiedot ja ohjelman yleiset tiedot. Tämä tiedosto toimii pikakuvakkeena käyttäjällä. Kun käyttäjä klikkaa OSD-tiedostoa, lähtee SoftGrid Client -asiakasohjelma hakemaan ohjelman SFT-tiedostoa palvelimelta, sekä käynnistää SystemGuard-säiliön.
- **ICO** – tiedosto on asennetun ohjelman ikonikuva, joka liitetään käyttäjälle OSD-tiedoston pikakuvakkeen ikoniksi, jotta ohjelma näyttäisi käyttäjälle normaalilta ohjelmalta.
- **SPRJ** (Sequencer Project File) – tiedosto on Sequencer-paketointiohjelman projektitiedosto, jonne on talletettu projektiin liittyvät tiedot. Tämän tiedoston avulla Sequencer-paketointiohjelmalla voidaan avata uudelleen virtualisoitu ohjelma esimerkiksi päivitystä varten.



Kuva 4.8: SoftGrid Sequencer-paketointiohjelman toiminta. Lähde: http://www.softricity.com

Ohjelmapaketoinnin neljä vaihetta SoftGrid Sequencer -paketointiohjelmalla (Kuva 4.8) (Softricity Sequencer 2005):

1. Ylläpitäjä käynnistää SoftGrid Sequencer -paketointiohjelman ja asentaa haluamansa ohjelman aivan kuin se asennettaisiin käyttäjän henkilökohtaiseen työasemaan. Sequencer-paketointiohjelma tutkii asennuksessa työasemalle syntyneet muutokset ja rakentaa niistä SystemGuard-säiliölle asetukset sekä paketoi ohjelman yhdeksi jaettavaksi tiedostoksi.

2. Ylläpitäjä käynnistää asennetun ohjelman testatakseen, että asennus on onnistunut. Samalla Sequencer-paketointiohjelma tutkii tarvitseeko ohjelma käynnistyäkseen käyttöjärjestelmän tiedostoja, joita ei asennuksessa havaittu ja ottaa tarvittaessa ne mukaan pakettiin.

3. Lopputuloksena Sequencer-paketointiohjelma tuottaa SoftGrid Data (SFT) -tiedoston, joka sisältää asennetun ohjelman tiedostot sekä SystemGuard-säiliön asetukset. Lisäksi Sequencer tuottaa myös XML-kuvaustiedoston, jota käytetään jaeltavan ohjelman jakelun hallintaan ja aktivointiin.

4. Sequencer-paketointiohjelman tuottamat tiedostot siirretään SoftGrid-palvelimelle, josta ohjelmaa jaellaan.

Sequencer-paketointiohjelma vaatii toimiakseen Windows 2000 käyttöjärjestelmän tai uudemman. Tämä myös asettaa vastaavan vaatimuksen virtualisoitavalle ohjelmalle, sen täytyy olla yhteensopiva vähintään Windows 2000 käyttöjärjestelmän kanssa. Sequencer tukee ohjelmasäiliöiden luontia seuraaville käyttöjärjestelmille:

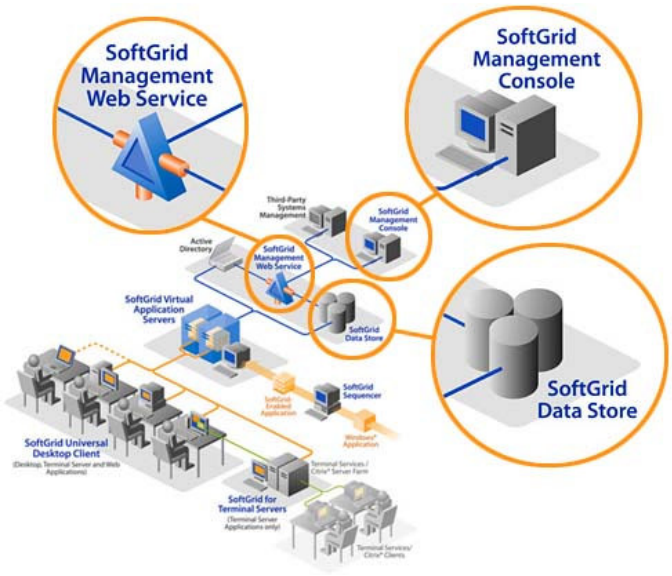
- Windows 2000 -tuoteperheen työasema- ja palvelinversiota
- Windows XP Pro
- Windows 2003 -tuoteperhettä

Sequencer-paketointiohjelmalla on myös tiettyjä rajoitteita tällä hetkellä. Sillä ei voida luoda toimivaa ohjelmasäiliötä ohjelmasta, joka asentaa palveluita tai ajureita. Henkilökohtaiset palomuuri- ja virustorjuntaohjelmistot ovat esimerkkisi tällaisia ohjelmia. Lisäksi ohjelmasäiliön koko ei saa kasvaa suuremmaksi kuin 4GB.

Yleisesti Softricity suosittelee käyttämään Sequencerin kanssa paketointityöasemassa juuri sitä käyttöjärjestelmäversiota, jolle ohjelmasäiliötä on tarkoitus jakaa. Kuitenkin useimmat ohjelmasäiliöt toimivat tästä seikasta huolimatta moitteetta eri käyttöjärjestelmäversioissa, koska ohjelmasäiliöt sisältävät tarvitsemansa tiedostot ja asetukset. Rajoite tulee ohjelman puolelta, kuten jo aiemmin totesin. Sequencerin tarkemmat vaatimukset löytyvät liitteestä 1.

4.1.5 SoftGrid-hallintakomponentit

SoftGrid -järjestelmän hallintaan kuuluu kolme komponenttia, *SoftGrid Data Store*, *SoftGrid Management Web Service* ja *SoftGrid Management Console* (Kuva 4.9).



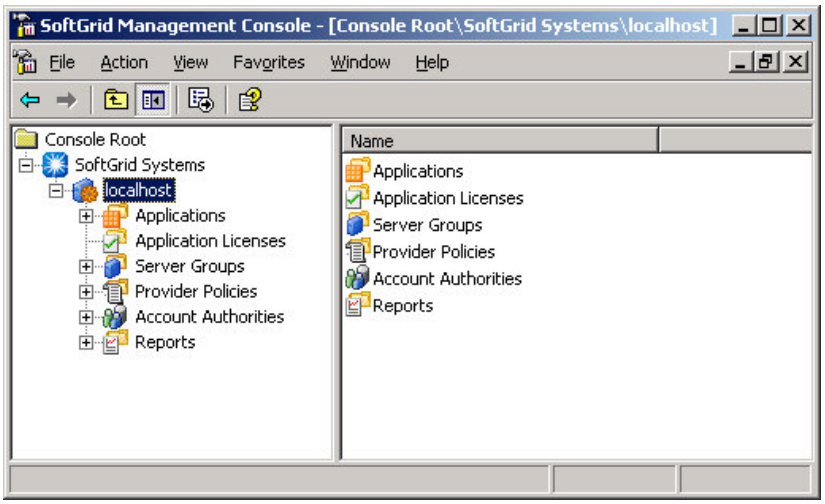
Kuva 4.9: SoftGrid-hallintakomponenttien sijoittuminen SoftGrid-järjestelmään. Lähde: http://www.softricity.com

SoftGrid Management Console

SoftGrid Management Console on SoftGrid-järjestelmän mukana tuleva MMC-täydennysosa (Microsoft Management Console). Se on SoftGrid-järjestelmän ylläpitäjälle näkyvin osa, koska sen avulla ylläpitäjä hallitsee koko SoftGrid-järjestelmää (Kuva 4.10).

SoftGrid Management Console tukee Windows 2000, XP ja 2003 käyttöjär jestelmiä eli se voidaan asentaa esimerkiksi ylläpitäjien työasemille. Soft-Grid Management Console:n tarkemmat vaatimukset löytyvät liitteestä 1.

SoftGrid Management Console on yhteydessä ainoastaan SoftGrid Mana-gement Web Service -komponenttiin, josta Management Console:n lähettä-mät käskyt ja pyynnöt lähetetään eteenpäin.



Kuva 4.10: SoftGrid Management Console.
Lähde: http://www.softricity.com

SoftGrid Management Console on velhopohjainen (Wizard-based) työkalu, jolla ylläpitäjä voi hallita seuraavia SoftGrid-järjestelmän toimintoja (Soft-ricity T&C 2004: 174 - 175):

- Jaettavat ohjelmat – Lisääminen, asetusten muuttaminen, poistami-nen ja listaaminen (Applications)
- Käyttäjien oikeudet jaeltaviin ohjelmiin (Applications)
- Lisenssit jaeltaviin ohjelmiin (Application Licenses)
- Raportit järjestelmästä ja sen käytöstä (Reports)
- Palvelimien ja palvelin ryhmien hallinta (Server Groups)
- Ylläpitäjien hallinta (Account Authoties)
- Säännöstöjen hallinta (Provider Policies)

Applications – haara sisältää luettelon julkaistuista ohjelmista ja niiden ti-loista (käytössä tai pois käytöstä). Tästä haarasta ylläpitäjä voi lisätä ja pois-taa ohjelmia luetteloon. Hän voi muokata seuraavia ohjelman asetuksia: palvelimet joille ohjelma on julkaistu, ohjelman lisenssiryhmä, ohjelman pikakuvakkeiden paikat, mihin tietyn päätteen omaavat tiedostot aukeavat ja käyttäjän tai käyttäjäryhmien oikeudet käyttää ohjelmaa. (Softricity T&C 2004: 174 – 175.)

Application Licenses – haara sisältää listan ohjelmille tarkoitetuista lisenseistä. Tästä haarasta ylläpitäjä voi lisätä ja poistaa ohjelmalisenssejä.

Lisenssejä on valittavana kolmea erilaista tyyppiä (Softricity T&C 2004: 174 – 175.):

- *Unlimited License* – Tämä lisenssityyppi sallii rajoittamattoman määrän yhteyksiä ohjelmaan, jolle tämän tyyppinen lisenssi annetaan.

- *Concurrent License* – Tämä lisenssityyppi sallii ainoastaan tietyn määrän samanaikaisia yhteyksiä ohjelmaan, jolle tämän tyyppinen lisenssi annetaan.

- *Named License* – Tämä lisenssityyppi sallii ainoastaan erikseen nimettyjen käyttäjien tai käyttäjäryhmien yhteydet ohjelmaan, jolle tämän tyyppinen lisenssi annetaan.

Kaikille näille lisensseille voidaan määrittää tietty päättymisajankohta sekä lisenssinumero, jonka avulla ylläpitäjät pystyvät pitämään yllä lukua lisenssiavaimista. (Softricity T&C 2004: 174 – 175.)

Osa lisenssityypeistä voidaan myös yhdistää toisiinsa, jotta niitä voitaisiin hallita vieläkin tarkemmin. Esimerkiksi Concurrent License voidaan yhdistää Named License -lisenssiin. Näin ylläpitäjällä on mahdollista myöntää viidenkymmenen hengen myyntiryhmälle oikeudet ainoastaan kymmeneen samanaikaiseen yhteyteen tiettyyn ohjelmaan. Tämä myös varmistaa, että ainoastaan viidelläkymmenellä myyntiryhmän jäsenellä on mahdollisuus käyttää tätä ohjelmaa. Lisenssiryhmät eivät ole ohjelmakohtaisia eli yhden lisenssiryhmän voi ottaa käyttöön mihin tahansa ohjelmaan, vaikka usein lisenssiryhmät tehdäänkin tiettyä ohjelmaa silmälläpitäen. (Softricity T&C 2004: 174 – 175.)

Oletuksena ohjelmien lisensointi on pois käytöstä ja se otetaan käyttöön säännöstöistä (Provider Policy). Valittavana on kahdenlaista lisensointitasoa (Softricity T&C 2004: 174 – 175):

- *Audit License usage only (oletus)* – Tällä tasolla ainoastaan kirjataan SoftGrid-tietokantaan lisenssien luovutus- ja palautustapahtumia, jotta ylläpitäjät voivat ajaa tapahtumista raportteja ja niiden perusteella tilata lisää tai vähentää lisenssejä tarpeen vaatiessa. Tämä taso ei siis estä ohjelmia toimimasta ilman lisenssiä.

- *Enforce License Policies* – Tällä tasolla lisensointi otetaan täysin käyttöön. Käyttäjä saa ohjelman käyttöönsä vain, jos ohjelmalle on määritelty käyttöön Unlimited License tai jos ohjelmalle on määritelty käyttöön Concurrent License ja yhteysmahdollisuuksia on vielä jäljellä tai jos ohjelmalle on määritelty käyttöön Named License ja

käyttäjä kuuluu nimettyihin henkilöihin, jotka saavat ohjelmaa käyttää.

Server Groups – haara sisältää palvelinryhmät ja ryhmien sisällä on yksittäiset palvelimet. Oletuksena SoftGrid-järjestelmän asennus luo Default Server Group -ryhmän, jonka alle ensimmäinen palvelin myös oletuksena liitetään. Softricity suosittelee käyttämään vain yhtä ryhmää ja vähintään kahta palvelinta. Kun palvelimet ovat samassa ryhmässä, voidaan niitä hallita yhtenä kokonaisuutena ja näin niiden verkkokuorma voidaan helposti jakaa. (Softricity T&C 2004: 145 – 150.)

Ylläpitäjä voi määrittää yhdelle palvelinryhmälle eri säännön (Provider Policy) kuin toiselle palvelinryhmälle. Palvelinryhmistä ylläpitäjä voi myös vaihtaa palvelinten tapahtumien kirjaamista (Logging). (Softricity T&C 2004: 145 – 150)

Yksittäisillä palvelimilla ylläpitäjät voivat vaihtaa muistin varausmäärää SoftGrid-järjestelmälle, palvelinten nimitietoja sekä liikennöintiprotokollia ja -portteja. (Softricity T&C 2004: 145 – 150.)

Provider Policy – haara sisältää SoftGrid-järjestelmän säännöt. Säännöissä voidaan määritellä millä tavalla käyttäjän varmennus tehdään, tietojen keräys käyttöasteista ja lisensoinnin käyttöönotto sekä lisensoinnin pakotus. SoftGrid-järjestelmä luo asennuksen aikana automaattisesti yhden valmiin säännön (*Default Provider Policy*), koska useimmissa tapauksissa ei ole tarvetta useammalle säännölle. Tämä sääntö on oletuksena liitetty Default Server Group -palvelinryhmälle. Useampia sääntöjä tarvitaan yleensä vain, jos ylläpitäjä haluaa lisensseillä rajoittaa muutamia tiettyjä ohjelmia. (Softricity T&C 2004: 152 – 155.)

Sääntöjä voidaan ottaa käyttöön kahdella eri tavalla, joko erillisesti jokaiselle ohjelmalle tai suoraan koko palvelinryhmälle. Softricity suosittelee ottamaan säännön käyttöön erikseen jokaiselle ohjelmalle. (Softricity T&C 2004: 152 – 155.)

Account Authorities – haara sisältää tiedot käytettävästä toimialueesta (Domain), kuten toimialueen tyypin (Active Directory tai Windows NT4 Domain) ja toimialueen nimen. Tästä haarasta pääsee myös vaihtamaan käyttäjätunnuksen ja salasanan tunnukselle, jota SoftGrid-järjestelmä tarvitsee tietojen lukemiseen toimialueelta. (Softricity T&C 2004: 155 – 156.)

Reports – haara sisältää järjestelmän raportointiominaisuudet. Raportointi on toteutettu Crystal Reports runtime -agentilla, joka koostaa raportit SoftGrid-tietokannasta (SoftGrid Data Store) joko tietyn päivän, viikon tai kuukauden ajalta. Raportit voidaan tuoda järjestelmästä PDF-tiedostoksi, ne voidaan tulostaa tai niitä voidaan katsoa näytöltä. (Softricity T&C 2004: 181.)

SoftGrid-järjestelmä sisältää seuraavia valmiita raporttipohjia (Softricity T&C 2004: 181):

- **Application Utilization** – Ylläpitäjä voi seurata eri ohjelmien käyttöasteita. Raportti näyttää myös käyttäjät, jotka ovat ohjelmaa käyttäneet.
- **License Compliance** – Ylläpitäjä voi seurata ohjelmille määriteltyjen lisenssien käyttöä.
- **Software Audit** – Ylläpitäjä voi seurata kaikkien ohjelmien käyttöasteita ja niiden käyttäjiä.
- **System Utilization** – Ylläpitäjä voi seurata yhden palvelimen, palvelinryhmän tai koko järjestelmän käyttöastetta.
- **User/Group Activity** – Ylläpitäjä voi seurata käyttäjän tai ryhmän käyttämiä ohjelmia ja milloin he ovat niitä käyttäneet.
- **System Error** – Ylläpitäjä voi seurata palvelinten aiheuttamia virheitä.

SoftGrid Management Web Service

SoftGrid Management Web Service on käytännössä WWW-palvelin, jonka tehtävänä on toimia komentojen tarkastajana sekä välittäjänä SoftGrid-järjestelmän ja SoftGrid Management Console -hallintatyökalun välissä. Tämä mahdollistaa, että järjestelmän saama komento tulee aina lailliselta taholta ja se on toimiva. Management Web Service hoitaa komentojen välityksen SoftGrid-tietokantaan ja Active Directory - tai Windows NT 4.0 -toimialueelle. (Softricity T&C 2004: 52.)

SoftGrid Management Web Service vaatii toimiakseen Microsoft IIS 5.0 tai uudemman WWW-palvelimen(Softricity T&C 2004: 52). Tarkemmat vaatimukset löytyvät liitteestä 1.

SoftGrid Data Store

SoftGrid Data Store on käytännössä SQL-tietokanta, jonne SoftGrid-järjestelmä varastoi dataa, pitää kirjaa tapahtumista ja hakee tietoa kaikista järjestelmälle oleellisista tiedoista. Näitä tietoja ovat järjestelmän säännöt, palvelinryhmät, palvelinten asetukset ja käyttäjien oikeudet. Lisäksi tietokantaan kirjataan tietoa ohjelmista ja lisensseistä sekä niiden käyttöasteista raportointia varten. Asennus luo tietokantaan tarvitsemansa taulut itse. (Softricity T&C 2004: 52.)

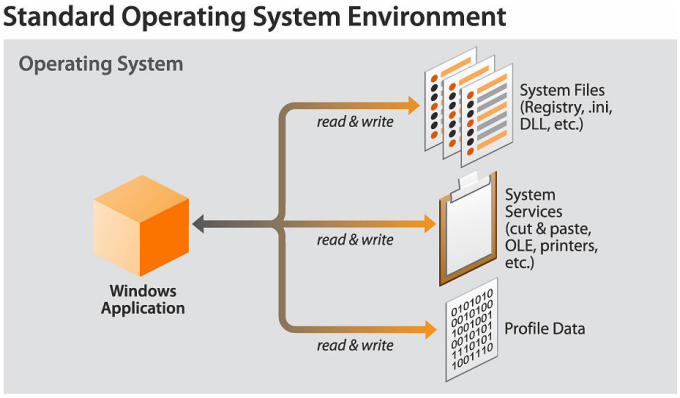
SoftGrid Data Store voidaan asentaa seuraaville tietokannoille (Softricity T&C 2004: 52.):

- MSDE: Microsoft SQL Server 2000 Desktop Engine
- SQL 2000: Microsoft SQL Server 2000

Softricity suosittelee käyttämään aina SQL 2000 tietokantaa. Tarkemmat vaatimukset löytyvät liitteestä 1.

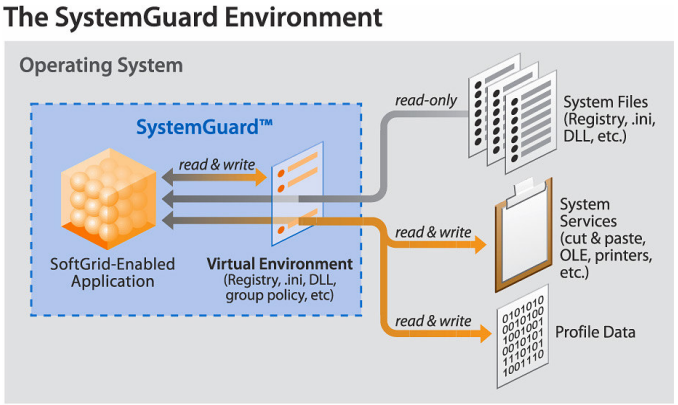
4.1.6 Virtualisoitu ohjelma

Kuten aiemmin kappaleessa *2.2 Ohjelmistojen asennusongelmat* todettiin, tavallisin menetelmin työasemaan asennettu ohjelma muuttaa käyttöjärjestelmää ja tulee kiinteäksi osaksi sitä. Sen poistaminen jälkiä jättämättä käyttöjärjestelmästä on hankalaa ja usein mahdotonta.



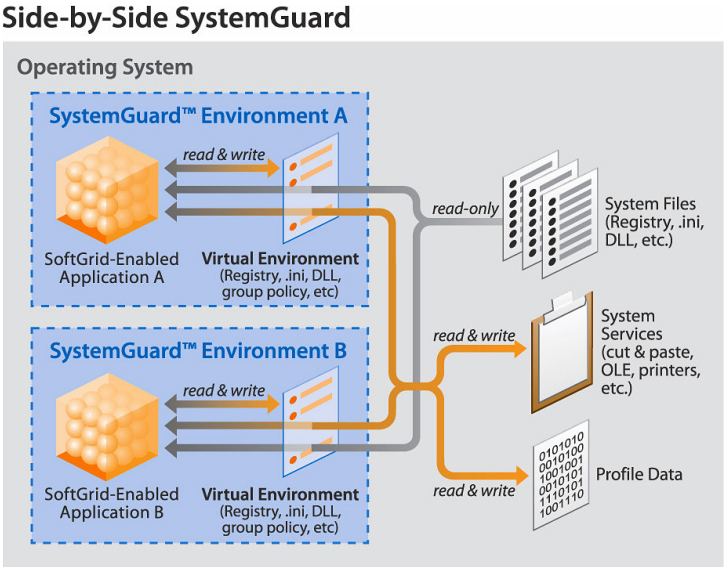
Kuva 4.11 Fyysisesti työasemaan asennettu Windows -ohjelma. Lähde: http://www.softricity.com

Kuvassa 4.11 on kuvattuna kuinka fyysisesti työasemalle asennettu ohjelma pääsee vapaasti kirjoittamaan asetuksensa käyttöjärjestelmään. Mikään ei estä ylikirjoittamasta toisten ohjelmien asetuksia. (Softricity Virtualization 2005).



Kuva 4.12: SoftGridillä virtualisoitu ohjelma.
Lähde: http://www.softricity.com

Kuvassa 4.12 on kuvattuna kuinka SystemGuard estää ohjelmaa vaikuttamasta missään vaiheessa työaseman paikalliseen tiedostojärjestelmään tai rekisteriin. Ohjelman sallitaan kirjoittaa rekisteriin tai tiedostojärjestelmään ainoastaan omassa säiliössään. Tästä huolimatta ohjelmalla on oikeus käyttää paikallisen työaseman prosessoria, muistia, verkkolevyjä ja muita resursseja hyväkseen aivan kuin perinteisesti asennettu ohjelma. (Softricity T&C 2004: 17.)

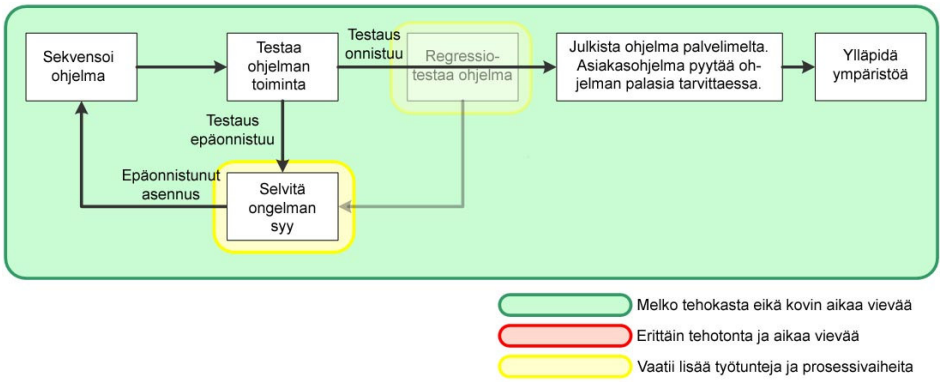


Kuva 4.13: Yhteiseen ohjelmasäiliöön virtualisoidut ohjelmat.
Lähde: http://www.softricity.com

Kuvassa 4.13 on kuvattu SoftGrid Suite eli ohjelmat, jotka omaavat yhteisen virtuaalisen ohjelmasäiliön. Näin ohjelmat, jotka vaativat erityisen tavan kommunikoida, pystyvät toimimaan ja kommunikoimaan toisilleen.

4.2 SoftGrid-tekniikan toimintamalli

Koska SoftGrid-tekniikka muuttaa ohjelmien asennusprosessia niin, ettei ohjelmia enää tarvitse erikseen fyysisesti asentaa jokaiselle koneelle, vaan ohjelmat jaetaan työasemille valmiiksi asennettuina säiliöinä. Tämä nopeuttaa ohjelmien käyttöönottoa. Toisin kuin ESD- ja SBC-tekniikalla, ylläpitäjän ei tarvitse enää tehdä pitkäkestoista regressio-testausta jokaisella työasemakokoonpanolla vaan riittää, että ohjelma on kertaalleen testattu toimivaksi, sillä se toimii jokaisessa koneessa samalla tavalla – itsenäisenä ohjelmana.



Kuva 4.14: SoftGrid-tekniikan toimintamalli. Lähde: Brown 2005: 14.

Kun tutkitaan kuvaa 4.14 ajankäytön ja hallinnoinnin kannalta, huomataan että prosessista on jäänyt pois aikaa kuluttavia vaiheita. Prosessista on tullut yksinkertaisempi. SoftGrid SystemGuard -säiliö eristää ohjelmat ja estää näin ristiriitojen syntymisen. Tämän ansiosta uusien ohjelmien levittäminen käyttäjien työasemille nopeutuu, koska prosessista on poistunut testausvaiheet, vianmääritysvaiheet ja SBC-tekniikan ohjelmistosiilot.

4.3 Ohjelmistojen hallintaprosessi SoftGrid-tekniikalla

Ohjelmien asennus, ylläpito, päivitys ja poisto SoftGrid-tekniikalla (Brown 2005: 16.):

- **Asennus** – Ohjelma asennetaan SoftGrid Sequencer -paketointiohjelman omaavalle työasemalle, jossa Sequencer-paketointiohjelma luo asennetusta ohjelmasta jaeltavan ohjelmasäiliön. Luotu ohjelmasäiliö kopioidaan SoftGrid-palvelimelle, jossa sille määritellään käyttäjien käyttöoikeudet ja mahdolliset lisenssit. Kun käyttäjä kirjautuu työasemalleen ja käynnistää ohjelman, pyytää SoftGrid Client -asiakasohjelma ohjelmaa palvelimelta, joka jakaa ohjelman käyttäjän työasemalle.

- **Ylläpito** – Ohjelmat käynnistyvät työasemilla omissa suojatuissa säiliöissään eli SoftGrid SystemGuard -säiliössä. Ohjelmat ovat erillään ympärillä olevista ohjelmista ja käyttöjärjestelmästä. Tästä syystä ohjelmat eivät myöskään tarvitse teknistä ylläpitoa erikseen, vaan jokainen ohjelma toimii samalla tavalla. Mikäli jaetussa ohjelmassa havaitaan virhe, on sama virhe kaikilla. SoftGrid-järjestelmän tapauksessa ohjelma voidaan korjata ja päivittää ilman katkoksia.

- **Päivitys** – Ohjelman päivitykset ja muutokset tehdään SoftGrid Sequencer -paketointiohjelmalla. Ohjelmasäiliö avataan Sequencer-ohjelmaan, joka purkaa ohjelmasäiliön työaseman kovalevylle ja palauttaa ohjelman samaan tilaan, kuin se olisi asennettuna työasemal-

la. Tämän jälkeen ylläpitäjä voi asentaa päivityksen ja/tai muuttaa ohjelman asetuksia. Kun päivitykset on tehty, Sequencer tarkastaa muutokset ja luo ohjelmasäiliön uudelleen. Päivitetty ohjelmasäiliö kopioidaan palvelimelle. Kun käyttäjä seuraavan kerran käynnistää ohjelman, huomaa SoftGrid Client -asiakasohjelma ohjelman muuttuneen palvelimella ja lataa ohjelman uuden version automaattisesti käyttäjän työasemalle.

- **Poisto** – Ylläpitäjä poistaa käyttäjältä oikeudet ohjelmaan joko SoftGrid-järjestelmästä tai toimialueen ryhmästä. Kun käyttäjä seuraavan kerran kirjautuu koneelleen, poistuu ohjelma ja ohjelman pikakuvake häneltä.

4.4 SoftGrid-tekniikan hyvät ja huonot puolet

Hyvät puolet:

- Ohjelmia ei asenneta tietokoneisiin fyysisesti.
- Ohjelmat, jotka aiheuttavat ristiriitoja, eivät enää tarvitse ohjelmistosiiloja tai erillistä tietokonetta, koska ohjelmat eivät enää pääse aiheuttamaan ristiriitoja.
- Siinä on keskitetty hallinta ohjelmien jakelulle, päivitykselle, tukemiselle ja poistolle.
- Ohjelman tiedostoja ja asetuksia ei koskaan ladata paikalliseen rekisteriin tai tiedostojärjestelmään, joten käyttöjärjestelmä pysyy puhtaana.
- Siinä on keskitetty käyttöoikeuksien ja lisenssien valvonta ohjelmakohtaisesti.
- Ohjelmat voidaan päivittää käytön aikana työasemille (mukaan lukien terminaalipalvelimet) ilman käyttökatkoja.
- Se vähentää huomattavasti ohjelmien ylläpitokustannuksia.
- Ylläpitäjän ei tarvitse erikseen varmistaa, että terminaalipalvelimella on asennettuna käyttäjän haluama ohjelma, sillä ohjelmat ovat käyttäjällä käytössä tunnustensa perusteella.
- SoftGrid on kypsä tuote, jolla on onnistuneesti luotu jo yli 20 000 erilaista ohjelmasäiliötä.

- Ohjelmasäiliöiden jakelu ja käyttö eivät vaadi käyttäjän kuulumista pääkäyttäjäryhmään, joten käyttäjien oikeuksia voidaan rajoittaa ja samalla tietoturvaa parantaa.

- Ohjelmista, käyttäjistä ja lisensseistä saadaan olemassa olevilla raporteilla seurattua monipuolisesti.

Huonot puolet:

- Ohjelmasäiliöiden luonti vaatii hyvää tuntemusta ohjelmasta käyttäjän näkökulmasta, jotta ohjelmista tulisi toimivasti esiasennettuja.

- Tällä hetkellä tuen puute ohjelmille, jotka:
    - asentavat ja tarvitsevat ajureita
    - käynnistyvät palveluna ennen käyttöjärjestelmää ja sisään kirjautumista
    - asentavat ja tarjoavat palveluja
- Ohjelmasäiliöiden koko on rajoitettu (4GB)

- Rajoitettu asiakaskäyttöjärjestelmä tuki (ei tukea Windows 95, 98, NT ja ME käyttöjärjestelmille).

5 Koulutusluokkaylläpidon ohjelmistohallinnan kehittämisprojekti SoftGrid-tekniikalla

Koulutusluokkaympäristö on otettu tähän opinnäytetyöhön mukaan, koska se toimii hyvänä esimerkkinä ohjelmistohallintaprosessin toiminnasta. Koulutusluokkaympäristössä näkyy koko ohjelmien elinkaari aina asennuksesta niiden poistoon, koska osa ohjelmista vaihtuu taajaan koulutuksen mukaan. Erona yritysten tuotantoympäristöihin usein on vain, että koulutusluokkaympäristöissä on uusimmat versiot ohjelmista aikaisemmin käytössä kuin tuotantoympäristöissä.

SoftGrid-järjestelmällä on mahdollista parantaa ohjelmien saatavuutta ja lisätä ohjelmien hallittavuutta niin koulutusluokkaympäristössä kuin yrityksen tuotantoympäristössä. Etuja, joita luokkaympäristön toimivasta SoftGrid-järjestelmästä voisi yrityksen tuotantoympäristöönkin heijastaa, ovat ainakin valmiiksi tuotetut ja testatut ohjelmapaketit. Mikäli ohjelmat ovat toimineet moitteetta koulutuskäytössä, toimivat nämä samat ohjelmapaketit myös tuotannon käytössä. Toisena etuna ovat yhtenäiset ohjelmien asennukset ja asetukset. Kun samoja ohjelmapaketteja voidaan käyttää koulutusympäristössä ja tuotantoympäristössä, muodostuu näistä ohjelmista molemmille ympäristöille yhtenäisiä. Toisin sanoen molemmista ympäristöistä tulee vakioituja ja inhimillisten virheiden määrä vähenee eikä asennusrutiinien eroista tarvitse huolehtia.

5.1 Projektiympäristön lähtökohdat

Koulutusluokan hallinta on toteutettu Altiris Client Management Suite -järjestelmällä, jolla Atea Finland Oy ylläpitää asiakkaan koulutusluokkien työasemia.

Altiris Deployment Solution for Clients -työkaluohjelmalla hoidetaan työasemien asennukset valmiilla levykuvauksilla eli image-tiedostoilla. Kaikista työasemien malleista on oma image, jota käytetään valmiina pohjana kaikille samaa mallia oleville työasemille. Tämä pohjamalli sisältää:

- valmiiksi asennetun käyttöjärjestelmän asetuksineen ja laiteajureineen
- yleiset toimisto-ohjelmat (Esim. Microsoft Office, Adobe Reader, jne.)
- muita tarpeellisia ohjelmia, jotka eivät häiritse toisia ohjelmia, eivätkä vaadi päivitystä kovin usein

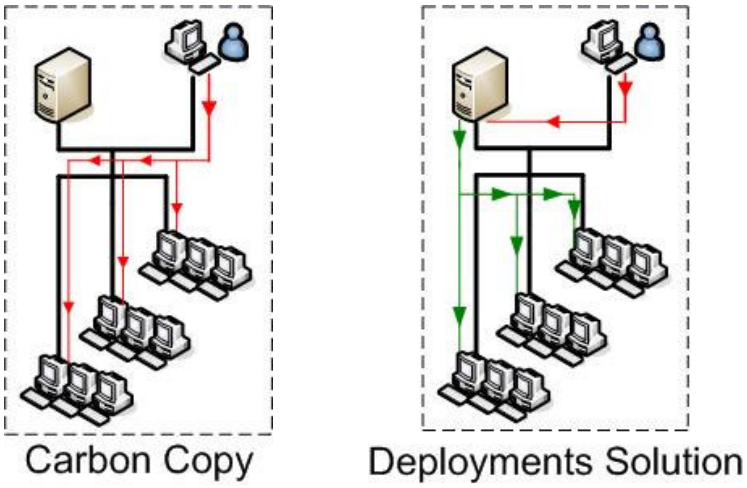
Käyttöjärjestelmän sekä ohjelmien päivitykset ovat pohjamallissa samalla tasolla kuin ne olivat sitä luotaessa eli mikäli pohjamalli on useampia viikkoja vanha, täytyy käyttöjärjestelmä ja ohjelmistot päivittää aina työaseman

asennuksen jälkeen. Tämä työosuus on automatisoitu IBM:n toteuttamalla Tivoli-hallintajärjestelmällä. Käyttäjän kirjautuessa työasemalla verkkoon, Tivoli tarkastaa käyttöjärjestelmän sekä ohjelmien nykyisen päivitystason ja päivittää ne Yritys X:n asettamalle tietoturvatasolle tarpeen niin vaatiessa. Näin ollen ei pohjamallien käyttöjärjestelmän ja ohjelmien päivityksistä tarvitse erikseen huolehtia.

Pohjamallit eivät kuitenkaan sisällä usein päivittyviä ohjelmia. Nämä ohjelmat täytyy aina asentaa ja päivittää erikseen kurssikohtaisesti jokaiselle työasemalle. Ohjelmat on myös poistettava erikseen jokaiselta työasemalta koulutuksen päätyttyä.

Erillisten ohjelmien asennukset ja päivitykset on toteutettu käyttäen Altiris-järjestelmän Carbon Copy -työkaluohjelmaa, joka mahdollistaa työasemien etähallinnan verkon kautta (Kuva 5.1). Tällä työkalulla ylläpitäjä voi asentaa, päivittää ja poistaa ohjelman työasemalta manuaalisesti käymättä työaseman luona fyysisesti.

Myös Altiris Deployments Solution for Clients -työkaluohjelmalla voidaan asentaa ohjelmia etäältä massa-asennuksina (Kuva 5.1). Tämä soveltuu ohjelmiin, jotka ovat Microsoft Installer (.MSI) tai vastaavalla tekniikalla asennettavia ohjelmia. Lisäksi Deployments Solution -työkalulla voidaan tehdä skriptipohjaisia ohjelma-asennuksia. Deployments Solution ei kuitenkaan ilmoita tapahtuneista asennusvirheistä ja ylläpitäjän pitää joka tapauksessa aina käydä erikseen testaamassa asennettujen ohjelmien toiminta jokaiselta työasemalta. Tähän testaamiseen käytetään Carbon Copy -etähallintaa.



Kuva 5.1: Esimerkki Carbon Copy ja Deployments solution for Clients

Altiris Carbon Copy ja Deployments Solution for Clients ovat yhdessä hyvä yhdistelmä, joiden avulla ylläpitäjät voivat työskennellä etäältä luokkien työasemien kanssa eivätkä pelkästään omassa vaan myös eri toimipisteissä.

Deployment Solution -työkalun tehtävät:

- käynnistää työasemat etäältä (Wake-On-LAN)
- asentaa työasemat uudelleen pohjamallista
- asentaa ohjelmia työasemille massa-asennuksena
- mahdollistaa etäyhteyden työasemiin ongelman selvitystä varten, mikäli Carbon Copy ei toimi tai sillä ei saada yhteyttä

Carbon Copy -työkalun tehtävät:

- toimii etäyhteysohjelmana koulutusluokkien työasemille
- mahdollistaa kaikki samat toimenpiteet, kuin fyysisessä työaseman käytössä

5.2 Projektiympäristön lähtötilanteen kehitystarpeet

Erilliset ohjelmien asennukset sekä poistot vievät paljon aikaa kurssien välissä ja tämä aika on aina pois kurssien aikataulusta. Tästä syystä olisi tärkeää saada ohjelma-asennuksen ajankäyttöä tehostettua ja ylläpitäjien aikaa vapautettua.

Ohjelmia on erikseen asennettava seuraavista syistä:

- Ohjelmat päivittyvät useasti vuoden tai joskus jopa kuukauden aikana.

- Osa erikseen asennettavista ohjelmista on lisäosia jollekin toiselle ohjelmalle eli ohjelmat täydentävät toisiaan ja ovat integroituja toisiinsa. Näitä lisäosia on erilaisia ja aivan kaikki niistä eivät toimi toistensa kanssa yhtäaikaisesti samalla työasemalla. Työasemilta on siis myös poistettava ohjelmia, jotka häiritsevät toisia ohjelmia.

- Joillekin ohjelmille on varattu vain muutamia lisenssejä ja tästä syystä nämä ohjelmat on aina poistettava, jotta lisenssit vapautuisivat ja ohjelmat voitaisiin asentaa toiseen koulutusluokkaan.

Ohjelmien erilliset asennukset aiheuttavat myös ongelman koulutusympäristöihin. Vaikka ohjelmien asennusvaiheet ovat dokumentoituja, ei asennuksista silti tule yhdenmukaisia johtuen eri alueiden asennusrutiineista ja tottumuksista. Asennusdokumentointi ei myöskään poista mahdollisuutta inhimillisiin virheisiin. Inhimillisten virheiden varalta ylläpitäjien on siis aina testattava asennetut ohjelmat jokaiselta luokan työasemalta erikseen.

5.3 Virtualisoinnilta odotetut ratkaisut luokkaylläpidolle

Johtuen SoftGrid-järjestelmän tavasta jaella ja käsitellä ohjelmia erillisinä sekä ympäristöstään riippumattomina komponentteina, odotetaan SoftGrid-järjestelmän tuovan luokkaylläpitoon seuraavia ratkaisuja:

Yhtenäiset ohjelmapaketit – Mahdollistavat saman ohjelman toimivuuden työaseman mallista riippumatta. Tällöin saadaan jätettyä pois erillisiä ohjelma-asennuksia ja niiden poistoja, joka vapauttaa ylläpitäjien aikaa ja mahdollistaa koulutusluokkaympäristön pysymisen alueesta riippumatta samanlaisena eli vakiona.

Helpottaa ohjelmien jakelua – Ylläpitäjien ei tarvitsisi enää asentaa ohjelmia erikseen vaan heidän tarvitsisi ainoastaan ladata valmiiksi asennetut ohjelmapaketit työasemille ennen kurssia keskitetyltä palvelimelta.

Ylläpitäjien huoleksi jäisi oikeanlaisen kurssiympäristön luonti ja poisto sekä koulutusluokkien työasemien, käyttöjärjestelmien ja tietoturvan ylläpito. Lisäksi mukaan ylläpitoon tulisi toimivien ohjelmapakettien luonti. Pois jäisivät ylimääräiset ohjelma-asennukset sekä niiden asetukset, jotka aiheuttavat eroavaisuuksia ja mahdollisuuden inhimillisiin virheisiin.

5.4 Projektiympäristön nykytilanne

Opinnäytetyön kirjoitushetkellä luokkaylläpidolle on rakennettu yksi SoftGrid-palvelin, joka sisältää SoftGrid-järjestelmän seuraavat komponentit: SoftGrid-palvelimen, -tietokannan ja WWW-palvelimen. Itse hallintakonsoli on asennettuna erillisellä työasemalla, jolla SoftGrid-järjestelmän toimintoja voidaan hallita etäältä, aivan kuten Altiriksen tapauksessa.

SoftGrid-järjestelmään on luotuna kolme erillistä jaettavaa ohjelmasäiliötä, joiden asennukset ovat olleet perinteisellä menetelmällä pitkäkestoisimmat ja hankalimmat. Ohjelmat on testattu toimiviksi eli käytännössä tämä tarkoittaa, että ohjelmat on jaettu onnistuneesti työasemille, ne ovat käynnistyneet työasemilla onnistuneesti sekä ohjelmilla on pystytty tekemään ne toimenpiteet mitä niillä on tarkoitus tehdä.

Näiden ohjelmien säiliöinti Sequencer-ohjelmalla ei kuitenkaan ole ollut mitenkään itsestään selvää. Hankaluuksia ohjelmasäiliöiden luonnissa on aiheuttanut juuri tietämättömyys kuinka eri käyttäjät ohjelmia käyttävät. Esimerkiksi ohjelmointiympäristöissä on useimmille käyttäjille tärkeää saada käyttöönsä komentokehote, jota kautta käyttäjät voivat vaihtoehtoisesti käyttää käännösohjelmia. Koska komentokehote on osa käyttöjärjestelmää ja ohjelmasäiliön idea on eristää ohjelma käyttöjärjestelmästä, on vaikea saada ohjelmasäiliö näkyviin komentokehotteelle. Lisäksi komentokehotteessa tulisi tietenkin olla kaikki tarvittavat ympäristömuuttujat oikeilla arvoilla. Nämä arvot ovat automaattisesti oikein graafisessa ohjelmointioh-

jelmassa, josta ohjelmasäiliö on alkuperäisesti tehty, mutta eivät kun tämä ohjelmasäiliö avataan komentokehotteen käyttöön.

Toisena hankaluutena voidaan pitää ohjelmasäiliöiden rajoitettua kokoa, sillä raskaimmat ohjelmointityökalut hyödyntävät usein monia eri ohjelmakääntäjiä sekä lisäkomponentteja ja tästä syystä ne pitäisi saada asennettua samaan ohjelmasäiliöön. Kun kaikki tarvittavat ohjelmat ovat samassa säiliössä, voi säiliön kokorajoitus tulla vastaan. Näin ollen osa ohjelmista ja komponenteista on asennettava fyysisesti työasemalle ja osa ohjelmasäiliöön. Haastavinta on löytää mitkä osat pitäisi jättää säiliöstä pois.

Yleisesti voidaan kuitenkin todeta, että SoftGrid-järjestelmällä ohjelmasäiliöiden luonti on kohtuullisen helppoa. Mitä monimutkaisempi on ohjelman asennus ja toiminta, sitä haastavampi on sen säiliöinti.

Ensimmäiset kurssit, joissa on käytetty virtualisoituja ohjelmia, on saatu päätökseen ja tulokset olivat hyviä. Ohjelmat toimivat niin kuin niiden pitääkin toimia. Ainoat ongelmat tulivat ns. *kahdennetusta asennuksesta*. Tällä tarkoitetaan, että ohjelmat olivat varmuuden vuoksi asennettuina myös fyysisesti koulutusluokan työasemille. Tämä aiheutti käyttäjille mahdollisuuden käyttää väärää versiota ohjelmasta, joka ei nähnyt käyttäjän luomia tiedostoja. Tämä asia korjaantuu, kun seuraavaksi siirrytään käyttämään pelkkiä virtualisoituja ohjelmia.

Tulevaisuudessa SoftGrid-järjestelmää on tarkoitus alkaa levittää useammalle eri alueelle ja saada se kattamaan kaikki toimipisteet. Näin saataisiin yhdistettyä kaikki koulutusluokat keskitetyn ohjelmistohallinnan alaisuuteen sekä yhtenäiset ohjelmistot niihin alueesta riippumatta. Myös ohjelmasäiliöiden määrä tulee kasvamaan tulevaisuudessa sitä mukaa kuin kursseja järjestetään ja ohjelmista saadaan luotua uusia ohjelmasäiliöitä. Tämä kaikki vaikuttaa myös palvelinkapasiteettiin, jota pitää myös kasvun myötä laajentaa ja turvata.

6 Pohdintaa

Ohjelmistovirtualisointi on tekniikkana tuore ja vielä melko tuntematon ainakin Suomessa, mutta Yhdysvalloissa ja Euroopassa Softricityn SoftGrid-järjestelmä on nostattanut jo melkoisesti mielenkiintoa ohjelmistovirtualisoinnin puolesta. Softricity kehittää SoftGrid-järjestelmää jatkuvasti. Mielestäni heillä on selkeä käsitys mihin suuntaan tuotetta tulisi viedä, sillä jokainen uusi versio on ottanut suuren askeleen eteenpäin edelliseen verrattuna.

Mielestäni ohjelmistojen virtualisoinnin tämän hetkinen heikkous on sen toimivuuden selkeä osoittaminen. Koska ohjelmistojen virtualisointi kääntää ajatusmallia ohjelmien fyysisestä asentamisesta pois päin ohjelmat on aina fyysisesti asennettava tavalla tai toisella tietokoneelle. Itselläni oli aluksi vaikeuksia hahmottaa kaikkia niitä hyötyjä ja mahdollisuuksia mitä itsenäiset ohjelmasäiliöt voisivat tarjota. Miten ohjelmat vain ilmestyvät itsestään työasemalle kirjautumisen yhteydessä ja miten ne seuraavat käyttäjää hänen kirjautuessaan toiselle työasemalle? Vasta kun itse sain fyysisesti nähdä ja käyttää SoftGrid-järjestelmää, ymmärsin mistä ohjelmistovirtualisoinnissa on kysymys. Toinen samaan heikkouteen liittyvä asia on yritykset, joilla on jo käytössään SBC-järjestelmä (Citrix Presentation Server tai Windows Terminal Server). Vaikka yritysten SBC-järjestelmissä olisikin ongelmia ja vaikka SoftGrid-järjestelmän pystyy näihin järjestelmiin ottamaan käyttöön ilman käyttökatkoja, on yrityksen IT-hankinnoista päättävät henkilöt vaikea saada näkemään SoftGrid-järjestelmällä saavutettavat edut ja uskomaan, että se myös toteuttaa ne. He eivät helposti lähde sijoittamaan uuteen järjestelmään, jos vanhassakin on vielä ongelmia.

Jos verrataan tekniikkana toisiinsa ohjelmistovirtualisointia sekä muita tässä tutkintotyössä esiteltyjä tekniikkoja, voidaan todeta, että niille kaikille on oma paikkansa ja käyttökohteensa. Käytännössä tämä tarkoittaa, että nämä tekniikat eivät ole suoranaisesti vertailukelpoisia vaan ne paremminkin täydentävät toisiaan. Esimerkiksi SoftGrid-järjestelmällä ei täysin voida syrjäyttää ESD-järjestelmää, sillä useimmat ESD-järjestelmät tarjoavat mahdollisuuden päivittää käyttöjärjestelmiä keskitetysti ja pitää yllä rekisteriä laitteistoista sekä asennuksista, toisin kuin pelkkä SoftGrid-järjestelmä. Samoin on SBC-järjestelmien kanssa. SBC-järjestelmä tarjoaa esimerkiksi yrityksille mahdollisuuden varmistaa, että yrityksen kriittisimmät ohjelmat ja niiden synnyttämä data pysyy yrityksen sisällä. Kun ohjelmat esitetään käyttäjille terminaalipalvelujen kautta, ei heidän omalle työasemalleen jää mitään jälkiä tästä ja ohjelmalla tuotettu data ei koskaan lähde ulos yrityksen sisäisestä verkosta, toisin kuin käyttämällä pelkkää SoftGrid-järjestelmää. Ohjelmistovirtualisoinnin tarkoitus onkin keskittyä ohjelmien aiheuttamien asennus- ja hallintaongelmien poistamiseen eli aiheeseen, johon muut tekniikat eivät ratkaisua tarjoa.

Itse tutkintotyön kirjoittaminen ohjelmistojen virtualisoinnista on ollut haastava prosessi, sillä aihe on todella tuore ja puolueetonta materiaalia aiheesta löytyy yllättävän vähän. Vaikka Softricity onkin julkaissut SoftGrid-järjestelmänsä jo 2000-luvun alussa, on mielenkiinto virtualisoituihin ohjelmiin lähtenyt voimistumaan vasta muutaman viimeisen vuoden aikana. Tämä luultavasti johtuu osittain virtualisoitujen käyttöjärjestelmien yleistymisestä sekä osittain Softricityn ja heidän yhteistyökumppaneidensa onnistuneesta markkinoinnista. Ohjelmistovirtualisointimarkkinoille on ilmestynyt Softricityn myötä muitakin valmistajia kuten Altiris tuotteellaan Software Virtualization Solution sekä Citrix on myös julkaisemassa omaa versiotaan aiheesta (Project Tarpon). Mielestäni ohjelmistovirtualisoinnin tulevaisuus on riippuvainen siitä kuinka selkeästi kyseisen tekniikan valmistajat saavat sen erottumaan muista virtualisointituotteista sekä kuka heistä saa luotua selkeimmän mielikuvan sen tarjoamista hyödyistä.

Päästyäni itse käyttämään ja kokeilemaan SoftGrid-järjestelmän toimintaa käytännössä, voin vain todeta, että ohjelmistohallinta on nyt vihdoin lähtenyt kulkemaan oikeaan suuntaan. Ajatus virtuaalisista ohjelmista vaikuttaa vähintään yhtä hassulta kuin muutama vuosi sitten ajatus virtuaalisista käyttöjärjestelmistä. Nyt tarkemmin ajatellen, virtuaaliset ohjelmat ovat luontainen jatko siihen mihin virtuaalisten käyttöjärjestelmien tarjoamat mahdollisuudet loppuvat. Esimerkiksi käyttöjärjestelmien virtualisoinnin kehittäjänä WMware ratkaisi useampien käyttöjärjestelmien rajoituksen samassa fyysisessä tietokoneessa tavalla, jota muuten ei olisi pystytty ratkaisemaan. Samaan asiaan ohjelmien osalta tähdätään ohjelmistovirtualisoinnilla. Ohjelmien asennus ja hallinta eivät juuri ole muuttuneet huolimatta tietotekniikan muusta kehityksestä. Ainakaan tällä hetkellä ei näköpiirissä ole mitään muuta edes lähelle samoja tuloksia pääsevää tekniikkaa kuin ohjelmistovirtualisointi.

Mietittyäni tämän tutkintotyön pohjalta tilanteita, joissa yritykset voisivat parhaiten tunnistaa SoftGrid-järjestelmän soveltuvuuden heidän käyttöönsä, listasin muutamia seuraavia esimerkkejä:

1. Ohjelmistojakelussa ilmenee seuraavia ongelmia:
   - Ohjelmajakelussa kuluu liikaa aikaa.
   - Ohjelmapakettien luonti on monimutkaista.
   - Puuttuu keino valvoa ohjelmien lisenssejä.
   - Ohjelmat muuttuvat tai päivittyvät usein.
   - Ohjelmien asennuksista ei tule vakioita.

2. Yrityksellä on tarve käyttää kahta ohjelmaa, jotka aiheuttavat toisilleen ristiriitoja samassa tietokoneessa.

3. Ylläpidon täytyy jatkuvasti käydä korjaamassa ohjelmia käyttäjien tai toisten ohjelmien tekemiltä muutoksilta.

4. Ylläpidolta kuluu paljon aikaa uuden ohjelman tai ohjelmapäivityksen testaamiseen.

5. Yrityksellä on tarve käyttää yhtä aikaa kahta eri versiota samasta ohjelmasta (esim. Java JRE tai Microsoft Access).

6. Terminaalipalvelinfarmeille on jouduttu tekemään ohjelmistosiiloja estämään ohjelmaristiriitoja.

7. Yrityksellä on käytössään ohjelmia, jotka eivät toimi terminaalipalvelimelta.

6.1 Kokemukset

Tässä opinnäytetyössä ei kuvata seikkaperäisesti ympäristöä, johon SoftGrid-järjestelmä on asennettuna, sillä Atean ja asiakasyrityksen välinen salassapitosopimus estää tämän. Esimerkiksi laitteistojen ja verkkorakenteen kuvaaminen on kielletty. Itse koen tärkeämpänä tämän opinnäytetyön lukijan kannalta, että työ on pidetty julkisena. Näin sitä voivat hyödyntää kaikki asiasta kiinnostuneet.

Mistä sain aiheen työlleni?

Kesällä 2005 kuulin ensimmäisen kerran SoftGrid-järjestelmästä ja kuinka tällä tuotteella voidaan muuntaa nykyisiä Windows-ohjelmia virtuaalisiksi. Asia kuulosti kiinnostavalta, joten halusin tietää lisää aiheesta. Samalla sain tietää kehitysprojektista, jossa SoftGrid-järjestelmää oli tarkoitus hyödyntää. Kun tähän projektiin oli tarve löytää toteuttaja, ilmaisin kiinnostukseni aiheesta ja samalla sain itselleni aiheen opinnäytetyötä varten.

Miten työni eteni?

Työni aiheen parissa alkoi tutkimalla, onko yleensä mahdollista toteuttaa kyseistä järjestelmää vai estääkö jokin nykyisen ympäristön tai SoftGrid-järjestelmän rajoitteista projektin toteuttamisen. Mitään varsinaista estettä järjestelmän käyttöönotolle ei löytynyt, joten kehitysprojekti päätettiin toteuttaa.

Koska SoftGrid-järjestelmä on tuotteena uusi, oli koulutus tähän aiheeseen välttämätön. Suoritin viikon mittaisen koulutuksen ja sertifikaatin SoftGrid-järjestelmästä. Samalla tilattiin SoftGrid-järjestelmään lisenssit sekä ohjelmistot.

Koulutuksen suoritettuani alkoi samalla opinnäytetyön kirjoitus. Koulutuksessa sain hyvän perustan järjestelmästä, joten opinnäytetyön tekninen osuus on koostettu koulutuksen ja sen materiaalin pohjalta. Koulutuksessa sain myös sopivia lähteitä opinnäytetyöni viitekehyksen rakentamiseen.

Opinnäytetyön kirjoittaminen yhdistettynä työelämään oli rankka kokemus ja välillä oli hetkiä, jolloin itse kehitysprojektin etenemiseen oli keskityttävä enemmän.

Kehitysprojekti on nyt saatu vaiheeseen, jossa SoftGrid-järjestelmän käyttöä on testattu ja saatu positiivisia tuloksia. Jatkossa SoftGrid-järjestelmän käyttö ja ylläpito dokumentoidaan sekä järjestelmän kehitys jatkuu. Samalla opinnäytetyöni on tullut vaiheeseen, jossa se mielestäni kertoo oleellisimmat asiat ohjelmistojen virtualisoinnista.

Minkälainen järjestelmästä tuli?

SoftGrid-palvelin – Käyttöjärjestelmäksi valittiin Windows 2003, sillä tarkoituksena on käyttää samaa palvelinta useampia vuosia ja vanhempi versio on kuitenkin päivitettävä uudempaan, kun se ei enää täytä asiakkaan tietoturvamäärityksiä. SoftGrid-järjestelmän komponenteista palvelimeen on asennettuna SoftGrid Management Web Service, SoftGrid Data Store sekä SoftGrid Virtual Application Server eli käytännössä Web Servicen pohjana toimii Microsoftin IIS -palvelu ja Data Storen pohjana Microsoft SQL 2000 Server. Palvelin liitettiin myös osaksi asiakkaan toimialuetta.

Hallintakonsoli – SoftGrid Management Console asennettiin erillisille työasemille, joista koulutusluokkien ylläpitäjät hallinnoivat luokkia.

Sequencer työasema – Järjestelmän asennuksen kannalta hankalin osuus on työasema, jolla ohjelmasäiliöitä luodaan. Koska jokainen ohjelmasäiliö tulisi aina rakentaa puhtaassa käyttöjärjestelmässä, on pystyttävä jokaisen luodun ohjelmasäiliön jälkeen palauttamaan työasema alkuperäiseen tilaan uuden ohjelmasäiliön luontia varten. Yksi tapa toteuttaa tämä olisi ollut tehdä levymalli eli image ja palauttaa näin työasema alkuperäiseen tilanteeseen. Päädyin kuitenkin tilamaan tähän tarkoitukseen lisenssin VMware Workstationiin, jonka avulla työasemassa voidaan virtuaalisesti käyttää yhtä aikaa useampia käyttöjärjestelmiä. Virtuaalisista käyttöjärjestelmistä voidaan ottaa *snapshot* eli palautuspiste, johon voidaan palata aina haluttaessa. Palaaminen palautuspisteeseen tapahtuu todella nopeasti. Näin työaseman palauttamisessa alkuperäiseen tilanteeseen säästetään huomattavasti enemmän aikaa kuin levymallista palauttamisessa. Lisäksi VMware mahdollistaa useamman käyttöjärjestelmäversion käytön yhtä aikaa, joten näin vältetään usean levymallin ylläpito eri käyttöjärjestelmille.

Mitä ongelmia järjestelmässä ilmeni?

Hallinnointi – Kuten aikaisemmin jo kerroin, ohjelmasäiliöiden jakelu perustuu toimialueen käyttäjien oikeuksiin. Jokaiselle ohjelmasäiliölle on luotava oma ryhmänsä toimialueelle ja lisättävä tähän ryhmään käyttäjiä tai toisia ryhmiä. Kun jaeltavia ohjelmia alkaa olla useita, on toimialueen oikeuskäytäntöjen hyvästä suunnittelusta suuri apu. Kun käytäntö on saatu suunni-

teltua toimivaksi, on siitä hyvä tehdä asianmukainen dokumentointi, jotta kaikilla ylläpitäjillä on samanlainen toimintaohje.

Toinen hallinnointiin liittyvä ongelma on jaeltavien ohjelmien käsittely SoftGrid-hallintakonsolin kautta. Ajan myötä hallintakonsoliin kertyy useita jaeltavia ohjelmia ja kaikki nämä ohjelmat ovat listattuna samassa ikkunassa allekkain. Tämä aiheuttaa helposti sekaannuksia, kun jaeltavien ohjelmien asetuksia halutaan muuttaa. Etenkin tilanteissa, jossa samasta ohjelmasta on tehty useampia erilaisia versioita, on vaikea löytää haluamansa ohjelma. Mielestäni tämän asian saisi korjattua siten, että hallintakonsoliin lisättäisiin puunäkymä, jossa ohjelmat ensin lajitellaan Suite (eli ohjelmistokokoelma) -kohtaisina ja näiden alta löytyisi kyseisen Suiten sisältämät ohjelmat.

Sekvensointi – Toimivien ohjelmasäiliöiden luonti on edellytys toimivalle kokonaisuudelle. Se on koko järjestelmän pääasia. Ilman koulutusta ei ohjelmasäiliöiden luonti onnistu. Ei riitä, että ohjelman osaa asentaa yksittäiselle työasemalle vaan lisäksi pitää tietää mitä ohjelma tarvitsee toimiakseen ja kuinka se SoftGrid-järjestelmässä saadaan toteutettua.

Ongelmiksi kehitysprojektissa muodostuivat Windows-käyttöjärjestelmän omat apuohjelmat ja niiden yhdistetty käyttö ohjelmasäiliöiden kanssa. Esimerkiksi, jos ohjelma kirjoittaa lokitiedostoa tiettyyn hakemistopolkuun, jota ei voi muuttaa toiseksi, syntyy lokitiedosto ohjelmasäiliön sisäiseen tiedostojärjestelmään. Ohjelmasäiliöiden idea on eristää ohjelma käyttöjärjestelmästä, joten tiedostohallinnan (Explorer) kautta ei tähän lokitiedostoon pääse suoraan käsiksi. Tiedostohallinta on Windows apuohjelma, joka liittyy syvästi itse käyttöjärjestelmään, eikä sitä voida virtualisoida. Tiedostoon on kuitenkin jollakin tavalla päästävä käsiksi ja yksi tapa on ohjelmasäiliötä luotaessa ottaa mukaan linkki työaseman fyysisellä kovalevyllä olevaan Explorer-ohjelmaan. Kun Explorer-ohjelma avataan linkkinä ohjelmasäiliöstä, saadaan samalla ohjelmasäiliön tiedostojärjestelmä avattua tiedostohallinnan käyttöön. Tämä on vain yksi esimerkkitilanne miksi on tärkeää hankkia koulutus, kun SoftGrid-järjestelmää ollaan ottamassa käyttöön.

Mikä merkitys SoftGrid-järjestelmällä on Atealle?

Rakennetun järjestelmän tavoitteena oli kehittää ohjelmistojen asennus- ja hallintatekniikkaa. Toisin sanoen tavoitteena oli poistaa ohjelmistojen hallintaprosessista ylimääräisiä ja toistoa aiheuttavia vaiheita sekä minimoida asennustekniset erot eri alueiden välillä. Kun edellä mainitut seikat pystytään toteuttamaan, saadaan ohjelmistojen hallintaprosessista kustannustehokkaampaa sekä aikasäästöjä ylläpitäjien työn osalta. Lisäksi ylläpitäjien työtä saadaan mielekkäämmäksi asennusrutiinien poistamisella.

Säästöt SoftGrid-järjestelmän osalta verrattuna aikaisempaan käytäntöön koulutusluokkaympäristössä ovat kiistattomat. Pääasiassa ne tulevat aikasäästönä ja ohjelmien uudelleen käytön mahdollisuudesta.

Esimerkki tehostuneesta ohjelmistojen hallintaprosessista koulutusluokkaympäristöissä SoftGrid-järjestelmän avulla:

Asennus – Aikaisemmin ylläpitäjän täytyi asentaa tietyt ohjelmat jokaiseen työasemaan erikseen. Ohjelmasta ja sen koosta riippuen aikaa yhteen ohjelmaan meni noin 10 – 30 minuuttia jokaista työasemaa kohden. Tämän jälkeen jokainen asennettu ohjelma oli vielä testattava toimivaksi. Kun tällä menetelmällä asennetaan ohjelma työasemaan, kuluu ylläpitäjältä kymmeneen työasemaan aikaa 1,5 - 5 tuntia.

Nykyisin SoftGrid-järjestelmällä samojen ohjelmien saaminen kymmenelle työasemalle kestää noin 10 – 15 minuuttia.

Päivitys – Aikaisemmin ylläpitäjän täytyi asentaa päivitykset tiettyihin ohjelmiin jokaiselle työasemalle erikseen ja testata niiden toiminta. Aikaa yhteen ohjelmaan meni noin 5 – 10 minuuttia eli kymmenellä työasemalla laskettuna noin 1 – 1,5 tuntia.

Nykyisin aikaa päivityksiin kuluu 5 – 10 minuuttiin, sillä päivitykset tehdään suoraan virtualisoituihin ohjelmasäiliöihin, joista ne ovat välittömästi käytettävissä kaikilla käyttäjillä.

Ylläpito – Aikaisemmin ylläpitäjän täytyi hakea syytä ohjelmistojen ongelmiin useasta eri paikasta käyttöjärjestelmästä. Ylläpitäjä ei koskaan voinut olla varma, että vika olisi juuri itse ohjelmassa tai sen asennuksessa vaan usein ratkaisu ongelmiin löytyi käyttäjän tekemästä muutoksesta esimerkiksi tiettyyn ympäristömuuttujaan, jolloin ohjelma ei toiminut oikein. Joissakin ongelmallisimmissa tapauksissa ei ole ollut muuta vaihtoehtoa kuin poistaa ohjelma ja asentaa se uudelleen.

Nykyisin ylläpitäjä voi poistaa ohjelmasäiliön työaseman välimuistista ja ladata ohjelman uudelleen työasemalle, jolloin kaikki virheelliset asetukset poistuvat sekä ylläpitäjän aikaa ei kulu juurikaan ongelman selvittelyyn.

Poisto – Aikaisemmin ylläpitäjän täytyi poistaa haluamansa ohjelmat jokaiselta työasemalta erikseen sekä testata, että työasemalle jääneet ohjelmat toimivat edelleen halutulla tavalla. Aikaa yhden ohjelman poistamiseen meni noin 5 – 10 minuuttia eli kymmenellä työasemalla tähän menisi aikaa noin 1 – 1,5 tuntia.

Nykyisin ylläpitäjän ei tarvitse kuin poistaa käyttäjältä käyttöoikeus kyseisiin ohjelmiin, johon aikaa ei kulu muutamaa minuuttia enempää.

Tästä syystä Atean kehitysprojektiin on valittu tuotteeksi SoftGrid, sillä millään toisella tämän hetkisellä järjestelmällä ei olisi voitu päästä samanlaiseen lopputulokseen.

Mitä mieltä olen työstäni?

Mielestäni aiheen valinta on onnistunut, sillä käsittelen aihetta, jota ei aiemmin ole opinnäytetöissä käsitelty. Lisäksi se on tarjonnut tekijälleen uusia haasteita ja antanut paljon mietittävää sekä mahdollisuuden oppia jotain aivan uutta. Kokonaisuutena olen työhöni tyytyväinen.

Jatkona tälle työlle voisi ajatella kuvausta kuinka RES PowerFuse ja RES Wisdom täydentäisivät SoftGrid-järjestelmää eli ajatuksena olisi, että koko käyttäjän ympäristö (työpöydät, asetukset, ohjelmat, jne.) pysyisi käyttäjälleen vakiona vaikka käyttäjä kirjautuisi aina eri työasemalle. Tämä olisi tietenkin keskitetysti hallittavissa.

Ohjelmistojen virtualisointi SoftGrid-järjestelmällä on vakuuttanut toiminnallaan minut niin positiivisesti, että tulevaisuudessa tulen varmasti toimimaan tämän tuotteen parissa ja seuraamaan sen kehitystä. Suosittelen siihen tutustumista kaikille aiheesta kiinnostuneille.

7 Lähteet

Brown, Douglas A. 2005. To Install or Not Install: Solving the Application Management Nightmare. Part II. [online][viitattu 26.10.2005].
http://www.dabcc.com/documents/Part II - To Install or Not Install - Rev 2.pdf

Madden, Brian 2004. To Silo or not to Silo: What application installation strategy is best for you? [online][viitattu 13.11.2005].
http://www.brianmadden.com/content/content.asp?ID=275

Pratschner, Steven 2001. Microsoft: Simplifying Deployment and Solving DLL Hell with the .NET Framework. [online][viitattu 4.11.2005].
http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndotnet/html/dplywithnet.asp

RFC 2326: Real Time Streaming Protocol (RTSP) [online][viitattu 6.11.2005].
http://rfc2326.x42.com/

RFC 2246: The TLS Protocol Version 1.0 [online][viitattu 6.11.2005].
http://rfc2246.x42.com/

Softricity Company 2005. Kotisivut. [online][viitattu 3.11.2005].
http://www.softricity.com/company/index.asp

Softricity Desktop 2005. Kotisivut. [online][viitattu 3.11.2005].
http://www.softricity.com/news/ecollateral/public/The-Softricity-Desktop.pdf

Softricity DoD 2005. Desktops On Demand. Kotisivut. [online][viitattu 3.11.2005].
http://www.softricity.com/news/ecollateral/public/Softricity-VMware-DoD-2005-09.pdf

Softricity Products 2005. Kotisivut. [online][viitattu 3.11.2005].
http://www.softricity.com/products/softgrid.asp

Softricity Sequencer 2005. Kotisivut. [online][viitattu 6.11.2005].
http://www.softricity.com/products/sequencer.asp

Softricity Training & Certification 2004. SoftGrid 3.2 Administration Training Guide.

Softricity Virtualization 2005. Kotisivut. [online][viitattu 6.11.2005].
http://www.softricity.com/products/virtualization.asp

Tietoa 2003. Tietotekniikan liiton lehti yhteisöjäsenille. [online][viitattu 12.11.2005].
http://www.tt-tori.fi/pls/ttl/docs/F1902557425/TIETOA 3_03.pdf

Wilson, Warren 2005. Virtual Applications: Booster Rockets for Dynamic Computing? [online][viitattu 13.11.2005].
http://www.softricity.com/news/ecollateral/public/summit.pdf

8 Liitteet

Liite 1: SoftGrid 3.2 -järjestelmän vaatimukset

SoftGrid Server

- Intel Pentium III 1GHz
- 1GB RAM per CPU (512MB minimum)
- 200MB available hard disk space (not including application storage)
- Microsoft Windows 2000 Server/Advanced Server or Windows Server 2003

SoftGrid Data Store

- Intel Pentium III 850MHz
- 512MB RAM
- 200MB available hard disk space
- Microsoft Windows 2000 Server/Advanced Server or Windows Server 2003
- MSDE or SQL Database Engine
- Active Directory Domain Controller or NT 4 PDC

SoftGrid Management Web Service

- Intel Pentium III 800MHz
- 256MB RAM
- 50MB available hard disk space
- Microsoft Windows 2000 Server/Advanced Server or Windows Server 2003
- Internet Information Service 5.0 or 6.0
- MDAC 2.6
- .NET Framework 1.1

SoftGrid Management Console

- Intel Pentium III 700MHz
- 128MB RAM
- Microsoft Windows 2000, XP, or 2003
- .NET Framework 1.1

SoftGrid Sequencer

- Intel Pentium III 850MHz
- 256MB RAM
- 500MB Page File
- Microsoft Windows 2000, XP, or 2003
- SCSI drives for OS and SoftGrid Sequencer software for maximum performance
- Any additional requirements needed by the applications and the OS

SoftGrid for Universal Desktop Client

- Intel Pentium III 700MHz
- 128MB RAM
- 10MB available hard disk space for installation + 2GB for cache
- Microsoft Windows 2000, XP, or 2003
- Any additional requirements needed by the applications and the OS

SoftGrid for Terminal Servers

- Intel Pentium III 850MHz
- 2GB RAM (minimum 256MB, actual dependent on number of users and applications)
- 10MB available hard disk space for installation + 4GB for cache
- 1GB Page File
- Microsoft Windows 2000 Server/Advanced Server or Windows Server 2003, with Terminal Services enabled
- Any additional requirements needed by the applications and the OS
- Separate SCSI drives for OS and SoftGrid software for maximum performance