JULKAISUSARJA

Ari Sivula

METAHAKEMISTON JA LDAP-HAKEMISTON ASENNUS, KONFIGUROINTI JA OHJELMOINTI SEINÄJOEN KOULUTUSKUNTAYHTYMÄLLE

SEINÄJOEN AMMATTIKORKEAKOULU

SEINÄJOEN AMMATTIKORKEAKOULUN JULKAISUSARJA
D. OPINNÄYTETÖITÄ 21

Ari Sivula

METAHAKEMISTON JA LDAP-HAKEMISTON ASENNUS, KONFIGUROINTI JA OHJELMOINTI SEINÄJOEN KOULUTUSKUNTAYHTYMÄLLE

SEINÄJOEN AMMATTIKORKEAKOULU
Seinäjoki 2006

SEINÄJOEN AMMATTIKORKEAKOULUN JULKAISUSARJA
Seinäjoki Polytechnic publications

A. TUTKIMUKSIA Research reports
B. RAPORTTEJA JA SELVITYKSIÄ Research papers
C. OPPIMATERIAALEJA Teaching materials
D. OPINNÄYTETÖITÄ Thesis

ISBN 952-5336-66-2 (PDF)
ISSN 1456-176X

TIIVISTELMÄ

Ari Sivula. 2006. Metahakemiston ja LDAP-hakemiston asennus, konfigurointi ja ohjelmointi Seinäjoen koulutuskuntayhtymälle. Seinäjoen ammattikorkeakoulun julkaisusarja D: Opinnäytetöitä 21, 97 s.

Tietojärjestelmien lisääntyessä käyttäjätunnus-salasanaparit lisääntyvät. Tämän myötä käyttäjähallinta vaikeutuu ja mahdollisten virheiden määrä kasvaa. Käyttäjät joudutaan tästä syystä kirjaamaan tarpeen vaatiessa eri hakemistojärjestelmiin. Käyttäjähallinnan parantamiseen on kehitetty erilaisia ratkaisuja. Hakemistot yhdistetään yleensä käyttäen metahakemistoa. Metahakemisto on hakemisto hakemistoista. Seinäjoen koulutuskuntayhtymällä on paljon erilaisia tietojärjestelmiä ja osaa niistä ei ole millään tapaa yhdistetty. Tämä aiheuttaa myös sen, että käyttäjien eri identiteettitiedot eivät ole kaikissa hakemistoissa välttämättä ajan tasalla.

Tämän opinnäytetyön tavoite on selvittää ja toteuttaa metahakemiston ja LDAP-hakemiston hallittu asennus, konfigurointi ja ohjelmointi Seinäjoen koulutuskuntayhtymän ympäristöön. Opinnäytetyössä on selvitys Microsoftin metahakemistotuotteesta (Microsoft Identity Integration Server 2003) sekä kahden eri elektronisen hakemiston toiminnasta. Opinnäytetyö kattaa kolmen eri hakemiston yhdistämisen metahakemistoon. Nämä hakemistot ovat aktiivihakemisto, LDAP-hakemisto (Active Directory Application Mode) ja opiskelijahallintojärjestelmä Winha. Opinnäytetyössä on kuvaus ympäristöstä, johon järjestelmä tullaan asentamaan. Järjestelmästä tulee osa suurta Seinäjoen koulutuskuntayhtymän ympäristöä. Järjestelmällä Seinäjoen koulutuskuntayhtymän ympäristössä mm. automatisoidaan käyttäjien luonti ja muokkaus eri hakemistojärjestelmiin. Tämä opinnäytetyö on konstruktiivinen. Opinnäytetyössä asennetaan, konfiguroidaan ja ohjelmoidaan uusi järjestelmä Seinäjoen koulutuskuntayhtymän tarpeeseen.

Metahakemiston ja LDAP-hakemiston asennus, konfigurointi ja ohjelmointi oli prosessina onnistunut. Asennus, konfigurointi ja ohjelmointi jakaantui yhteentoista eri vaiheeseen. Tuloksena on selkeä esitys metahakemiston asennuksesta, konfiguroinnista, ohjelmoinnista ja kolmen eri hakemiston yhdistämisestä metahakemistoon Seinäjoen koulutuskuntayhtymän ympäristössä. Opinnäytetyöstä hyötyvät vastaavanlaisen ympäristön omaavat organisaatiot. Kaikkiaan prosessia voidaan pitää onnistuneena, koska asennukset, konfiguroinnit ja ohjelmoinnit olivat onnistuneita sekä alussa asetetut tavoitteet saavutettiin.

Asiasanat: Aktiivihakemisto, Keskitetty käyttäjähallinta, LDAP-hakemisto, Metahakemisto

ABSTRACT

Ari Sivula. 2006. The installation, configuration and programming of a metadirectory and an LDAP directory for Seinäjoki Joint Municipal Authority for Education. Publications of Seinäjoki Polytechnic D: Thesis 21, 97 p.

When information systems are multiplying, user passwords and usernames are multiplying too. This leads to difficulties in user management and it may also cause more mistakes. For this reason administrators have to register users to different directory systems separately. Different services and programs have been developed for the betterment of the user management. Directories are commonly connected to each other with a metadirectory. The metadirectory is the directory of directories. Seinäjoki Joint Municipal Authority for Education has many different kinds of information systems and some of them haven't been connected to each other in any way. This leads to a situation where the user's identity information is not necessary up to date in all directories.

The aim of this thesis is to solve and carry through the managed installation, configuration and programming of the metadirectory and an LDAP directory in the environment of Seinäjoki Joint Municipal Authority for Education. The thesis explains the activities of the Microsoft metadirectory product and two different kind of electronic directories. The thesis covers the connection of three different directories to the metadirectory. These directories are the active directory, an LDAP directory and the student administration system Winha. The system will automate the creation and forming of new users in different directory systems. This thesis is constructive. The thesis covers the installation and configuration of the new system for the needs of Seinäjoki Joint Municipal Authority for Education.

The result of the thesis is the clear demonstration of the installation of the metadirectory and three different directories combined to it. An organisation which has the same kind of environment will benefit from the thesis. All in all this process can be considered successful, because the installation, configuration and programming were a success and in the beginning directed goals were achieved.

Keywords: Active Directory, Centralized user management, LDAP directory, Metadirectory

Sisällys

Kuviot

Taulukot

Esimerkit

KÄYTETYT TERMIT JA LYHENTEET

ADAM = Active Directory Application Mode. Microsoftin kehittämä LDAP-hakemistotuote.

CSV-tiedosto = Comma Separated Values. Fyysinen ASCII-tiedostorakenne, joka sisältää puolipisteillä eroteltuja arvoja.

DLL-kirjasto = Dynamic Link Library. Kirjasto, jossa on suoritettavaa ohjelmistokoodia ja se voidaan ladata ajokelpoisesta sovelluksesta käsin.

DNS = Domain Name Service. Palvelu, joka muuntaa IP-osoitteet selkokieliseksi.

Domain Controller = Palvelintietokone, joka pitää yllä toimialueen infrastruktuuria. Aktiivihakemisto toimii Domain Controller -palvelimella.

Identiteettitiedot = Tiedot, jolla henkilö voidaan tunnistaa.

IP-osoite = IP-osoite (Internet Protocol) on numerosarja, jolla tietokoneet tunnistetaan verkossa.

LDAP-hakemisto = LDAP-protokollalla (Lightweight Directory Access Protocol) toimiva elektroninen hakemisto.

Metahakemisto = Hakemisto eri hakemistoista, eli siihen yhdistetään erilaisia elektronisia hakemistoja.

MIIS 2003 = Microsoft Identity Integration Server 2003. Microsoftin kehittämä metahakemistotuote.

Provisiointi = Tapahtuma, jossa luodaan tai muokataan objekteja kohdehakemistoon.

Replikointi = Prosessi, jossa tehdään identtinen kopio jostakin tiedosta.

Skeema = Joukko sääntöjä dokumentin/objektin rakenteelle ja sisällölle.

SQL Server 2000 = Microsoftin kehittämä SQL-palvelintuote, joka on versiota 2000.

Synkronointi = Prosessi, jossa yhdennetään yhtä tai useampaa tietoa samaksi.

Toimialue = Joukko palvelimia, tietokoneita ja laitteita, jotka jakavat saman ryhmänimen.

Winha = WM-datan kehittämä opiskelijahallintojärjestelmä.

(Webopedia, [viitattu 7.10.2005])

1 JOHDANTO

Tietojärjestelmien lisääntyessä käyttäjätunnus-salasanaparit lisääntyvät. Organisaatioissa on yleensä monia eri tietojärjestelmiä ja jokaiseen tietojärjestelmään käyttäjillä on eri käyttäjätunnus-salasanapari. Nykyään organisaatiot pyrkivät pääsemään niin sanottuun keskitettyyn käyttäjähallintaan. Tämä tarkoittaa mm. sitä, että jokaiseen organisaation tietojärjestelmään käyttäjillä on vain yksi käyttäjätunnus-salasanapari. Metahakemisto tarvitaan juuri tällaiseen tilanteeseen. Seinäjoen koulutuskuntayhtymällä on tämä tilanne. Tietojärjestelmiä on paljon ja tarvitaan keskittämistä.

Tämän opinnäytetyön tavoite on selvittää ja toteuttaa metahakemiston hallittu asennus, konfigurointi ja ohjelmointi Seinäjoen koulutuskuntayhtymän ympäristöön. Metahakemiston hyöty on se, että käyttäjien identiteettitiedot pysyvät samanlaisina jokaisessa hakemistojärjestelmässä. Opinnäytetyöstä hyötyvät myös vastaavanlaisen tietoverkon omaavat organisaatiot. Opinnäytetyö rajataan koskemaan Seinäjoen koulutuskuntayhtymän tilannetta.

Opinnäytetyössä selvitetään Microsoftin LDAP-hakemistotuotteen (Active Directory Application Mode) ja Microsoftin aktiivihakemiston (Active Directory) toiminta. Opinnäytetyössä on selvitys Microsoftin metahakemistotuotteesta (Microsoft Identity Integration Server 2003). Opinnäytetyö kattaa myös muiden hakemistojen, kuten Microsoftin aktiivihakemiston yhdistämisen metahakemistoon.

Opinnäytetyössä selvitetään Seinäjoen koulutuskuntayhtymän tilanne nyt ja paras mahdollinen lopputulos metahakemiston asennuksen, konfiguroinnin ja ohjelmoinnin jälkeen. Opinnäytetyössä on selvitys Seinäjoen koulutuskuntayhtymän EPEDU.local-toimialueesta.

Toteuttajan kannalta mielenkiintoa projektiin toi Seinäjoen koulutuskuntayhtymän todellinen tarve kyseiselle järjestelmälle, oma mielenkiinto aiheeseen ja atk-henkilöstön aktiivinen osallistuminen projektiin. Projektin aikana pidettiin monia kokouksia liittyen metahakemistoon ja LDAP-hakemistoon. Kokoukset veivät projektia eteenpäin ja lopputulos tarkentui kokousten myötä. Kokouksien päätökset ja pääkeskusteluaiheet ovat liitteessä 1. Tieto myös siitä, että järjestelmä otetaan varsinaiseen käyttöön, toi mielekkyyttä prosessiin.

1.1 Tutkimusongelma, aiheen rajaus ja ratkaisumenetelmä

Tällä hetkellä Seinäjoen koulutuskuntayhtymän selvänä ongelmana on se, että on olemassa monia eri hakemistopalvelimia, ja kaikki palvelimet eivät synkronoi/repli-

koi tietoja keskenään. Tämä aiheuttaa sen, että identiteettitiedot joudutaan kirjamaan eri hakemistoihin tarpeen mukaan käsin ja tiedot eivät ole jokaisessa hakemistossa ajan tasalla. Tämä nousee ongelmaksi mm. silloin, kun käyttäjät tarvitsevat monia eri tietojärjestelmiä ja käyttäjän tiedot vaihtuvat. Käyttäjillä on myös tästä johtuen monta eri käyttäjätunnus-salasanaparia. Selvä ongelma on myös se, että käyttäjähallintaan ei ole automatiikkaa. Seinäjoen koulutuskuntayhtymän ympäristöstä puuttuu kokonaan varsinainen metahakemisto ja LDAP-hakemisto.

Opinnäytetyön keskeiseksi ongelmaksi nousee se, millä saadaan eri hakemistopalvelimet synkronoimaan identiteettitiedot keskenään ja millä LDAP-hakemiston ja metahakemiston asennus, konfigurointi ja ohjelmointi saadaan tehtyä hallitusti. Tähän tarpeeseen on kehitetty erilaisia metahakemistotuotteita ja LDAP-hakemistotuotteita. Opinnäytetyön ongelmana on myös käyttäjätunnuksen luominen automaattisesti.

Metahakemistotuotteeksi valittiin Microsoft Identity Integration Server 2003 ja LDAP-hakemistotuoteeksi valittiin Microsoftin Active Directory Application Mode. Opinnäytetyö on konstruktiivinen. Konstruktiiviselle tutkimukselle on luonteenomaista uuden todellisuuden rakentaminen olemassa olevan (tutkimus)tiedon pohjalta (Järvinen, P. & Järvinen, A. 2002, 102). Konstruktiivisen tutkimuksen vaiheet ilmenevät tässä opinnäytetyössä seuraavasti. Ensin selvitetään nykytilanne hakemistojärjestelmien osalta. Tämän jälkeen tehdään suunnitelma asennuksista, konfiguroinneista ja ohjelmoinnista. Sitten tehdään asennukset, konfiguroinnit ja ohjelmoinnit suunnitelman mukaan ja testataan järjestelmää testiympäristössä. Näiden vaiheiden jälkeen suoritetaan järjestelmän varsinainen käyttöönotto.

Opinnäytetyö rajattiin koskemaan Seinäjoen koulutuskuntayhtymän tilannetta. Opinnäytetyössä perehdytään kolmen eri hakemiston yhdentämiseen. Nämä hakemistot ovat Microsoftin aktiivihakemisto (Active Directory), Microsoftin LDAP-hakemistotuote Active Directory Application Mode sekä WM-datan opiskelijahallintojärjestelmä Winha. Muita hakemistoja, jotka tulevat lopulliseen metahakemistoon, ovat WM-datan henkilöstöhallintojärjestelmä ja Datamarin opiskelijahallintojärjestelmä Studenta, mutta ne eivät ole osa tätä opinnäytetyötä. Opinnäytetyö ei perehdy Microsoft Windows 2003 Server -palvelinkäyttöjärjestelmän asennukseen eikä aktiivihakemiston asennukseen, koska Seinäjoen koulutuskuntayhtymällä on jo valmiiksi asennetut aktiivihakemistopalvelimet (Domain Controller). Seinäjoen koulutuskuntayhtymällä on kahdenlaisia hakemistoja, toiset ovat paperilla ja toiset ovat elektronisessa muodossa. Tässä opinnäytetyössä perehdytään elektronisessa muodossa oleviin hakemistoihin. Opinnäytetyön lopputuloksena on selkeä esitys metahakemistosta, kolmesta eri hakemistosta ja niiden yhdistämisestä metahakemistoon Seinäjoen koulutuskuntayhtymän ympäristössä.

Tässä opinnäytetyössä käsitellään metahakemistoa ja kolmen eri hakemiston yhdistämistä metahakemistoon. Opinnäytteessä käsitellään metahakemiston ja LDAP-hakemiston asentamista, konfigurointia ja ohjelmointia Seinäjoen koulutuskuntayhtymän tarpeen mukaisesti. Metahakemistotuoteeksi valittin Microsoft Identity Integration Server 2003. Hakemistot, jotka yhdistetään metahakemistoon tässä opinnäytetyössä, ovat Microsoftin aktiivihakemisto, Microsoftin LDAP-hakemistotuote Active Directory Application Mode ja WM-datan opiskelijahallintojärjestelmä Winha. Opinnäytetyössä rakennetaan testiympäristö, jolla voidaan testata järjestelmän toimintaa. Testiympäristö on samantapainen kuin Seinäjoen koulutuskuntayhtymän varsinainen ympäristö. Opinnäytetyön päättyessä ei kuitenkaan koko projekti ole loppuunviety. Seinäjoen koulutuskuntayhtymän koko projekti jatkuu vielä tämän opinnäytetyön jälkeen, koska muitakin hakemistoja tullaan vielä yhdistämään metahakemistoon. Opinnäytetyön jälkeen metahakemistoon tullaan yhdistämään WM-datan henkilöstöhallintojärjestelmä ja Datamarin opiskelijahallintojärjestelmä Studenta. Tämän työn tuloksesta jatkaminen on suhteellisen helppoa, koska loput yhdistettävät hallintojärjestelmät ovat samanlaisia kuin opiskelijahallintojärjestelmä Winha.

Lopputuloksesta hyötyvät Seinäjoen koulutuskuntayhtymän atk-henkilöstö, henkilökunta ja opiskelijat. Käyttäjähallinta paranee huomattavasti, koska ei tarvitse kirjata samaa käyttäjää eri hakemistoihin. Virheiden määrä vähenee tämän myötä. Käyttäjän identiteettitiedot pysyvät samoina ja ajan tasalla eri hakemistoissa. Tietoverkon käyttäjät eivät tarvitse kuin yhden käyttäjätunnus-salasanaparin Seinäjoen koulutuskuntayhtymän ympäristöön ja eri järjestelmiin. LDAP-hakemiston hyöty on mm. se, että kaikkien eri tietojärjestelmien (esim. Moodle) käyttäjätodennukset voidaan hoitaa LDAP-hakemistosta. LDAP-hakemistoon voidaan myös tallentaa paljon muutakin tietoa kuin käyttäjän tiedot (esim. kurssitietoja).

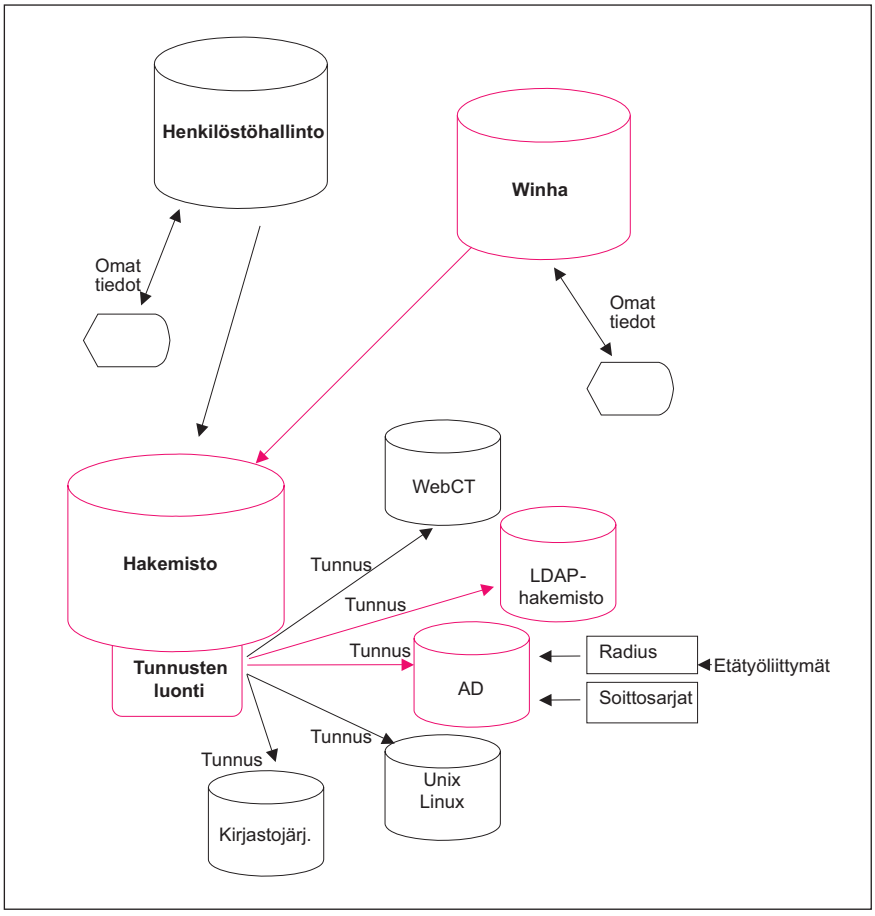
Termi- ja lyhenneluetteloon on kerätty tässä opinnäytetyössä keskeisimmät hakemistoihin liittyvät termit ja lyhenteet. Opinnäytetyössä on muitakin termejä ja lyhenteitä. Termeistä ja lyhenteistä kerrotaan enemmän niiden ilmetessä tekstissä myöhemmin.

1.2 Projektin päätarkoitus ja opinnäytetyön tavoite

Haka-projekti (hakemistot käyttäjähallinnossa) on yliopistojen, ammattikorkeakoulujen atk-keskusten ja tieteen tietotekniikan keskuksen CSC:n yhteinen projekti. Haka-projektin päämäärä on saattaa korkeakoulujen käyttäjähallinta sellaiselle tasolle, että korkeakoulut voivat luottaa toisiinsa käyttäjähallinnan näkökulmasta. Haka-infrastruktuuriprojekti on alkanut alun perin vuonna 2002. Seinäjoen koulutuskuntayhtymä tuli projektiin mukaan syksyllä 2003. Seinäjoen koulutuskuntayhtymällä oli tarve mm. saada käyttäjätunnusten luonti automatisoitua. Tämä ominaisuus on yksi metahakemiston tärkeimmistä. Tämän opinnäytetyön tekijä tuli projektiin mukaan keväällä 2005.

Suomen korkeakoulujen käyttäjähallinnan tilanne on useimmissa korkeakouluissa melkein sama kuin nykyinen Seinäjoen koulutuskuntayhtymän tilanne. Seinäjoen koulutuskuntayhtymän nykytilanne käyttäjähallinnassa on kuvattu ympäristön kuvauksessa luvussa 2. Suomen korkeakoulut yrittävät kuitenkin päästä keskitettyyn käyttäjähallintaan, koska se nopeuttaa asioita ja säästää aikaa. Muutamissa Suomen korkeakouluissa on käytössä erilaisia keskitetyn käyttäjähallinan ratkaisuja. Esimerkiksi Tampereen ammattikorkeakoulussa on käytössä Sun Java Enterprise System. Sun Java Enterprise System on Sun:in kehittämä palvelu käyttäjähallintaan.

Seinäjoen koulutuskuntayhtymän projektin tavoite on saada kaikki hakemistojärjestelmät synkronoimaan keskenään, tunnusten luonti automatisoitua ja käyttäjähallinta keskitettyä. Synkronoitavat hakemistot ovat aktiivihakemisto, LDAP-hakemisto, Winha, henkilöstöhallintojärjestelmä ja Studenta. Tavoite on lisäksi käyttäjätunnusten luonnin automatisointi LDAP-hakemistoon (Active Directory Application Mode) ja aktiivihakemistoon. Kuviossa 1 on kuvattu Seinäjoen koulutuskuntayhtymän mahdollinen tavoitetila koko projektin jälkeen.



Kuvio 1. Tavoitetilanne koko projektin jälkeen.
Lähde: Myllyaho, A., sähköpostikeskustelu 7.10.2005

Kuviossa 1 on kuvattu kuinka käyttäjätunnukset virtaavat hakemistoista toisiinsa. Kuviossa on yksi päähakemisto, johon tiedot viedään Winhasta ja Henkilöstöhallinnosta. Tämän jälkeen tunnukset jaetaan eri hakemistoihin. Radius-protokolla (Remote Authentication Dial-In User Service) ja soittosarjat käyttävät aktiivihakemistoa (AD) hyödykseen kirjautumisessa. Nykytilanteessa käyttäjätunnuksia ei luoda automaattisesti vaan käsin. Kaikki tiedot eivät ole tästä johtuen ajan tasalla. Uusi järjestelmä ratkaisee mm. tämän ongelman. Kuviossa 1 on merkittynä punaisella tässä opinnäytetyössä käsiteltävät hakemistot. Tämän opinnäytetyön puitteessa käsitellään Winhan yhdistämistä päähakemistoon (metahakemistoon) ja tunnusten synkronointia kahteen eri hakemistojärjestelmään.

1.3 Tutkimusraportin rakenne

Raportin toisessa luvussa käsitellään ympäristöä, johon järjestelmä asennetaan. Toisessa luvussa on kuvaus mm. Seinäjoen koulutuskuntayhtymän aktiivihakemiston nykyisestä rakenteesta. Raportin toisessa luvussa on myös kuvaus hakemistojen nykytilanteesta.

Kolmannessa luvussa käsitellään tarkemmin Microsoftin metahakemistotuotteen (Microsoft Identity Integraton Server 2003) toimintaa. Luvussa on lisäksi selvitys Microsoft Identity Integration Server 2003:n eri osa-alueista. Kolmannessa luvussa on selvitys SQL Server 2000:n roolista Microsoft Identity Integration Server 2003:n toiminnassa.

Neljännessä luvussa perehdytään kahteen eri elektroniseen hakemistoon. Nämä hakemistot ovat Microsoftin aktiivihakemisto ja Microsoftin LDAP-hakemistotuote Active Directory Application Mode. Neljännessä luvussa on selvitys mm. näiden hakemistojen arkkitehtuurista ja toiminnasta.

Viidennessä luvussa perehdytään metahakemiston ja LDAP-hakemiston asennukseen, konfigurointiin ja ohjelmointiin. Tässä luvussa on mm. selvitetty Microsoft Identity Integration Server 2003:n konfigurointi ja ohjelmointi Seinäjoen koulutuskuntayhtymän tilannetta varten.

Kuudennessa luvussa on selvitys järjestelmän testauksesta, käyttöönotosta ja niiden etenemisestä. Tässä luvussa on raportoitu testauksessa havaitut virheet ja niiden korjaukset.

Seitsemännessä luvussa on arvioita ja johtopäätöksiä opinnäytetyöstä ja sen onnistumisesta.

2 YMPÄRISTÖN KUVAUS JA NYKYTILANNE

Seinäjoen koulutuskuntayhtymässä on työntekijöitä eri yksiköiden palveluksessa tällä hetkellä noin 900 henkilöä. Opiskelijoita Seinäjoen koulutuskuntayhtymässä on tällä hetkellä noin 8600 henkilöä. Käyttäjiä ympäristössä on tästä johtuen paljon. Seinäjoen koulutuskuntayhtymään kuuluvat seuraavat oppilaitokset ja yksiköt:

- Seinäjoen ammattikorkeakoulu
  - Informaatio- ja kommunikaatioteknologian yksikkö Seinäjoella
  - Kulttuurialan ja muotoilun yksikkö Jurvassa
  - Liiketalouden yksikkö Seinäjoella
  - Maa- ja metsätalouden yksikkö Ilmajoella ja Ähtärissä
  - Ravitsemisalan yksikkö Kauhajoella
  - Sosiaali- ja terveysalan yksikkö Seinäjoella
  - Tekniikan yksikkö Seinäjoella
  - Yrittäjyyden yksikkö Kauhavalla

- Seinäjoen koulutuskeskus
  - Seinäjoen ammattioppilaitos
  - Seinäjoen palvelualojen oppilaitos
  - Kauhajoen palvelualojen oppilaitos
  - Maatalous- ja metsäoppilaitos
  - Taidon ja kulttuurin oppilaitos TAIKU
  - Seinäjoen koulutuskeskus, aikuiskoulutus
  - Seinäjoen koulutuskeskus, oppisopimuspalvelut

- Yhteiset palvelut
  - Kielipalveluyksikkö
  - Seinäjoen ammattikorkeakoulun teknologia- ja yrityspalvelukeskus SeiTek
  - Seinäjoen ammattikorkeakoulun sosiaali- ja terveysalan tutkimus- ja kehitystoiminnan yksikkö SoTe
  - Korkeakoulukirjasto
  - SC-Research Lapualla.
    (SeAMK - yksiköt, [viitattu 26.10.2005]; Sedu, [viitattu 26.10.2005])

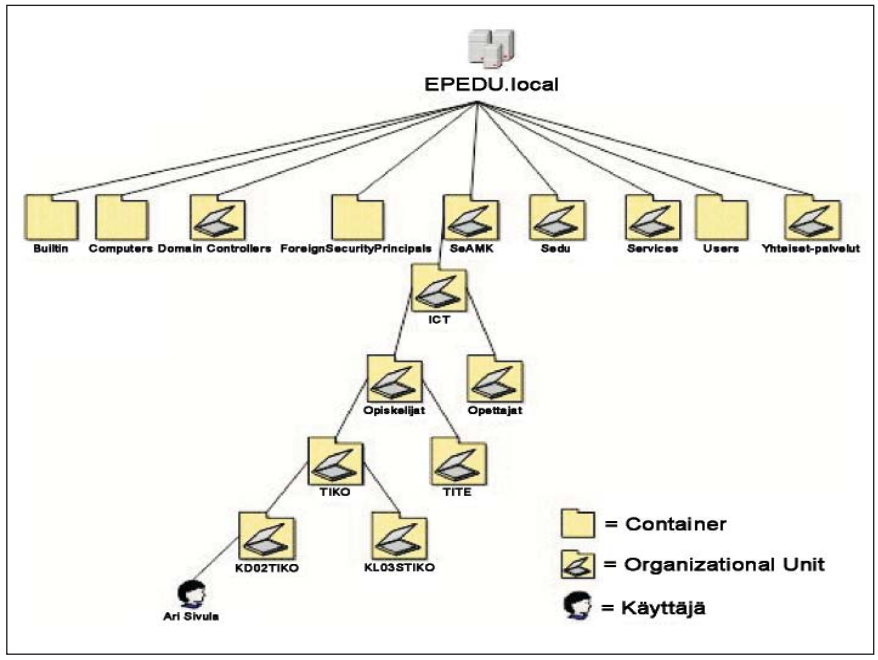
Seinäjoen koulutuskuntayhtymän organisaatiokaavio on kuvattu liitteessä 2. Järjestelmästä tulee osa Seinäjoen koulutuskuntayhtymän suurta ympäristöä. Seinäjoen koulutuskuntayhtymän verkkoon (EPEDU.local) kuuluu tietokoneita noin 3110, näistä palvelimia on noin 80. Tietokoneet ja palvelimet sijaitsevat eri yksiköissä ja paikkakunnilla. Palvelimista hakemistopalvelimia on noin 13 kappaletta, joista Domain Controllereita 8 kappaletta. Palvelimissa käyttöjärjestelminä ovat Microsoft Windows Server 2003, Microsoft Windows Server 2000, Microsoft Windows NT ja eri Linux-versioita. Jokaisella palvelimella on oma roolinsa koulutuskuntayhtymän verkossa. (Mäkelä, V-M., haastattelu 7.9.2005)

Seinäjoen koulutuskuntayhtymän verkko on pääosin Ethernet-pohjainen, johon kuuluu mm. useita tietokoneita, palvelimia, reitittimiä ja kytkimiä. Ethernet-nopeus vaihtelee yksikön sijainnista riippuen. Seinäjoella sijaitsevien Informaatio- ja kommunikaatioteknologian, Liiketalouden, Sosiaali- ja terveysalan ja Tekniikan yksiköiden nopeus on 100 megabittiä sekunnissa. Kulttuurialan ja muotoilun (Jurva), Maa- ja metsätalouden (Ähtäri ja Ilmajoki), Ravitsemisalan (Kauhajoki) ja Yrittäjyyden (Kauhava) yksiköiden nopeus on 4 megabittiä sekunnissa. (Mäkelä, V-M., haastattelu 7.9.2005)

Käyttäjätunnusten määrä tietoverkossa on noin 10250. Ylläpitäjiä tietoverkossa on tällä hetkellä 32. Ylläpitäjistä atk-pääsuunnittelijoita on 6, atk-suunnittelijoita on 13 ja atk-tukihenkilöitä on 13. Kaikki käyttäjätunnukset ovat aktiivihakemistossa ja henkilöiden identiteettitiedot löytyvät myös muistakin hakemistoista, kuten opiskelijahallintojärjestelmä Winhasta tai henkilöstöhallintojärjestelmästä. (Mäkelä, V-M., haastattelu 7.9.2005)

Eri ohjelmistoja ympäristössä on noin 60 kappaletta, joista eri käyttöjärjestelmiä on 6. Ohjelmistopaketteja ovat mm. Microsoft Office 2003, Microsoft Visual Studio .Net 2003 sekä monet muut ohjelmistot ja ohjelmistopaketit. (Mäkelä, V-M., haastattelu 7.9.2005)

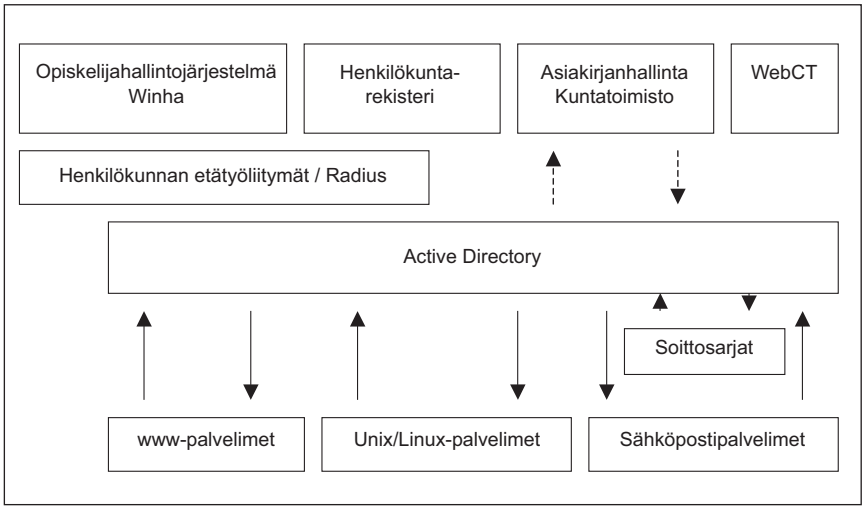
Seinäjoen koulutuskuntayhtymän aktiivihakemiston juuri ja osa sen rakenteesta on kuvattu kuviossa 2. Koko rakennetta ei kuviossa ole kuvattu, koska aktiivihakemiston rakenne on suuri.



Kuvio 2. Seinäjoen koulutuskuntayhtymän aktiivihakemiston juuri.

Kuviossa 2 on kuvattu Seinäjoen koulutuskuntayhtymän aktiivihakemiston juuri. Kuviossa 2 on myös esimerkki missä yksi opiskelija sijaitsee aktiviihakemiston objektihierarkiassa. Objektihierarkia rakentuu eri Organizational Uniteista. Organizational Unitit auttavat pitämään hakemistorakenteen loogisena. Objektihierarkian tulee olemaan samantyylinen myös LDAP-hakemistossa.

Seinäjoen koulutuskuntayhtymän nykytilanteessa tietojärjestelmiä on paljon ja suurinta osaa niitä ei ole yhdistetty millään tavalla. Metahakemisto ja LDAP-hakemisto puuttuvat Seinäjoen koulutuskuntayhtymältä kokonaan. Kuvio 3 esittää Seinäjoen koulutuskuntayhtymän nykytilannetta.



Kuvio 3. Nykytilanteen kuvaus. Lähde: Myllyaho, A., sähköpostikeskustelu 7.10.2005

Kuviossa 3 on kuvattu tietojärjestestelmät ja niiden väliset yhteydet. Varsinaisia yhteyksiä hakemistojen välillä ei ole, koska metahakemistoa ei ole nykytilanteessa ollenkaan. Kuviossa 3 on kuvattu, mitkä palvelimet autentikoivat käyttäjät aktiivihakemistosta (Active Directory). Kuviossa 3 on myös kuvattu ne järjestelmät, jotka eivät ole yhteydessä minkään hakemiston kanssa.

Metahakemisto ja LDAP-hakemisto auttavat nykytilannetta paljon käyttäjähallinnan näkökulmasta. Metahakemisto luo uudet käyttäjäobjektit aktiivihakemistoon ja LDAP-hakemistoon. Jos muutoksia tulee primäärilähteisiin (esim. Winha), niin muutokset synkronoituvat aktiivihakemistoon ja LDAP-hakemistoon. Tämä nopeuttaa käyttäjähallintaa, koska muutoksia ei tarvitse kirjoittaa jokaiseen tietojärjestelmään erikseen. Eri järjestelmien autentikoinnit voidaan suorittaa LDAP-hakemistosta. Tämä helpottaa käyttäjähallintaa, koska eri järjestelmät autentikoivat käyttäjät samasta hakemistosta. Metahakemisto ja LDAP-hakemisto helpottavat paljon käyttäjähallintaa Seinäjoen koulutuskuntayhtymän tilanteessa suuren käyttäjämäärän takia.

3 MICROSOFT IDENTITY INTEGRATION SERVER 2003

Organisaatiossa on yleensä monia eri elektronisia hakemistoja, kuten LDAP-hakemisto, aktiivihakemisto ja monet muut hakemistojärjestelmät. Organisaatioissa on yleensä ongelmia juuri tämän takia. Koska on paljon käyttäjiä ja paljon tietojärjestelmiä, niin käyttäjähallinta huononee ja vaikeutuu. Microsoft Identity Integration Server 2003 (MIIS 2003) on kehitetty käyttäjähallintaan. MIIS 2003 on metahakemistopalvelu, jolla on mahdollista yhdentää hakemistojen identiteettitiedot samankaltaisiksi. MIIS 2003 varastoi omaan kantaansa (Metaverse) identiteettitiedot. MIIS 2003 auttaa edistämään tuottavuutta, alentamaan tietoturvariskejä ja käyttäjähallintaan meneviä kustannuksia. Metahakemisto on niin sanottu hakemisto hakemistoista, eli siihen yhdistetään hakemistoja. Tässä luvussa on tietoa Microsoftin kehittämästä metahakemistotuotteesta. Microsoft Identity Integration Server 2003 valittiin Seinäjoen koulutuskuntayhtymän ympäristön metahakemistotuotteeksi. Valinta oli Seinäjoen koulutuskuntayhtymällä helppo ja looginen, koska suurin osa tietokoneista ja palvelimista on Microsoftin käyttöjärjestelmillä varustettuja. Seinäjoen koulutuskuntayhtymällä on käytössä Microsoftin aktiivihakemiston lisäksi muitakin Microsoftin tuotteita.

3.1 Historia

Microsoft Identity Integration Server 2003 on kolmas suurempi julkaisu Microsoftin kehittämästä identiteettihallintasovelluksesta. Identiteettihallinnalla tarkoitetaan käyttäjähallintaa, joka on kaikkea sitä millä saadaan käyttäjien tiedot pidettyä ajan tasalla eri järjestelmissä. Identiteettihallintasovelluksen saattaminen nykyiseen muotoonsa on vaatinut yli kahdeksan vuotta kehitystyötä.

Microsoft osti alun perin identiteettihallintasovelluksen heinäkuussa 1999 yritykseltä nimeltä ZoomIt. ZoomIt:n identiteettihallintasovelluksen nimi oli VIA 2.1. ZoomIt oli kehittänyt jo jonkin aikaa identiteettihallintasovellustaan. ZoomIt julkaisi ensimmäisen versionsa VIA 1.0 lokakuussa vuonna 1996. Oston jälkeen joulukuussa vuonna 1999 Microsoft nimesi uudestaan identiteettihallintasovelluksen ja julkaisi siitä uuden version. Uusi nimi oli Microsoft Metadirectory Services 2.1 (MMS 2.1). Seuraavan version, jonka nimi oli Microsoft Metadirectory Services 2.2, Microsoft julkaisi heinäkuussa vuonna 2000. Vuonna 2001 Microsoft julkaisi Service Pack:in MMS 2.2:lle, joka antoi vakautta identiteettihallintasovellukselle. MMS 2.2 SP1 oli viimeisin julkaisu ennen kuin MIIS 2003 julkaistiin. Vaikka MIIS 2003 oli julkaistu, se tunnettiin kuitenkin nimillä MMS 2003 ja MMS 3.0. MIIS 2003:n ensimmäinen Service Pack julkaistiin vuonna 2004. Microsoft julkaisi MIIS 2003:n rinnalla pelkistetymmän version MIIS 2003:sta ilmaiskäyttöön. Pelkistetymmän version nimi on Identity Integration Feature Pack 1a (IIFP). MIIS

2003 SP1 on identiteettihallintasovelluksen viimeisin versio. Seuraava Service Pack (SP2) julkaistaan vuonna 2006. (Microsoft Identity Integration Server 2003 Home, [viitattu 10.6.2005])

3.2 MIIS 2003 -metahakemiston versiokehitys

ZoomIt:n VIA toimii Windows NT -palvelimissa ja Windows 95 -käyttöjärjestelmissä. VIA koostuu kolmesta eri osasta

- varsinaisesta hakemistopalvelimesta
- ZoomIt Compass asiakasohjelmistosta
  - Ohjelmalla pystytään näyttämään ja muokkaamaan metahakemiston sisältöä.
- eri Management Agenteista.

VIA tukee muutamia eri email-, ohjelmisto- ja verkkokäyttöjärjestelmä-palvelinhakemistoja, kuten esimerkiksi Lotus Notes, Windows NT toimialueet ja Novellin Directory Services. VIA toimi replikointiperiaatteella, eli se ei synkronoinut tietoja. (Zoomit Via and the Dedicated Metadirectory, [viitattu 19.7.2005])

Microsoft Metadirectory Services toimii kahdessa eri Windows 2000 -palvelimessa, Windows 2000 Advanced Server tai Windows 2000 Datacenter Server. MMS tukee muutamia hakemistoja enemmän kuin VIA. MMS tukee seuraavia hakemistoja

- Banyan Vines
- Lotus cc:Mail
- Lotus Notes
- Active Directory
- Microsoft Exchange (LDAP- ja MAPI-pohjainen tuki)
- Microsoft NT
- Netscape LDAP
- Novell Groupwise
- Novell NDS (LDAP-pohjainen tuki)
- Novell Netware.

MMS:n ensimmäisestä versiosta lähtien hakemistot synkronoitiin keskenään. (Metaconnections, [viitattu 15.7.2005])

Microsoft Identity Integration Server 2003 toimii Windows 2003 Enterprise -palvelimissa. MIIS 2003 vaatii myös Microsoft SQL Server 2000 Standard tai Enterprise Editionin. SQL Server 2000:n on oltava asennettuna Service Pack 3. MIIS 2003 käyttää SQL Server 2000 -tietokantaa pääkantanaan. MIIS 2003 osaa integroitua Microsoft Visual Studio .Net 2003:n kanssa. Rules Extensioneita ohjelmoidessa tämä ominaisuus on todella käytännöllinen. Rules Extensionit ovat ohjelmoitavia laajennuksia MIIS 2003:een. Rules Extensioneista kerrotaan enemmän luvussa

3.3.4. MIIS 2003:ssa on tuki moneen eri hakemistoon. Management Agentit tukevat seuraavia hakemistoja

- Active Directory
- Active Directory Application Mode (ADAM)
- Attribute-value pair -tekstitiedostot
- Comma Separated Value -tiedostot
- Delimited -tekstitiedostot
- Directory Services Markup Language (DSML) 2.0
- Exchange 5.5 ja Exchange 5.5 Bridgehead
- Exchange 2000 ja Exchange 2003 Global Address List (GAL) -synkronointi
- Fixed-width -tekstitiedostot
- IBM DB2 ja IBM Tivoli Directory Server
- LDAP Directory Interchange Format (LDIF)
- Lotus Notes/Domino 4.6/5.0/6.0
- Novell eDirectory
- Sun/iPlanet/Netscape directory 4.x/5.x (“changelog”-tuki)
- Microsoft SQL Server 2000 ja SQL Server 7.0
- Microsoft Windows NT 4 -toimialueet
- Oracle 8i/9i
- Informix, dBase, ODBC ja OLE DB-tuki SQL Serverin Data Transformation -palveluille.

(Technical Overview of MIIS 2003 with Service Pack 1, 2005)

Microsoft Identity Integration Feature Pack 1a on tuotteena muuten ominaisuuksiltaan ja vaatimuksiltaan samanlainen kuin MIIS 2003, mutta tuettuja hakemistoja on vähemmän. Tuettuja hakemistoja ovat

- Active Directory
- Active Directory Application Mode (ADAM)
- Microsoft Exchange 2000 Server
- Exchange Server 2003 eri toteutukset.

3.3 MIIS 2003 -metahakemiston toiminta

MIIS 2003 koostuu pääosin viidestä eri objektista:

- Datalähteistä, joita voi olla monta tai muutama. Yleensä datalähteet ovat hakemistoja.
- Management Agenteista, joka on jokaisella hakemistolla omansa.
- Connector Spacesta, jossa jokaisella hakemistolla on oma tallennusalueensa.
- Metaversestä.
- Rules Extensioneista.

Jokaisella näistä objekteista on oma tärkeä roolinsa synkronointiprosessin aikana. MIIS 2003:n tietovirrat ovat kuvattu kuviossa 4.

3.3.1 Management Agentit

Management Agentit linkittävät datalähteitä MIIS 2003:een. Management Agentit ovat vastuussa tiedon siirrosta datalähteiden ja MIIS 2003:n välillä. Dataa muokattaessa MIIS 2003:ssa Management Agentit voivat viedä datan yhdistettyyn datalähteeseen ja synkronoida tiedon samaksi. Tällä tavoin tiedot pysyvät samoina MIIS 2003:ssa ja datalähteessä. Jokaisella datalähteellä on ainakin yksi Management Agent.

3.3.2 Connector Space

MIIS 2003:n Connector Space on tallennusalue, jota Management Agentit hyödyntävät tiedon tuontiin ja vientiin MIIS 2003:n ja datalähteen välillä. Jokaisella datalähteellä on oma looginen tallennusalueensa Connector Spacessa, jota kyseisen datalähteen Management Agent pitää yllä.

Connector Space on perimmiltään peilikuva yhdistetystä datalähteestä, jossa jokaisella objektilla on vastaava merkintä Connector Spacessa. Connector Space ei sisällä yhdistetyn datalähteen objekteja, vaan joukon objektin attribuutteja, jotka ovat määritetty Management Agentille.

Connector Space sisältää kaikki attribuutit, jotka ovat merkitty yhdistetystä datalähteestä, tietokannasta tai tiedostosta. Silloin, kun identiteettitieto on saatavilla Connector Spacesta, niin organisaatiot voivat helposti selailla identiteettitietoa koskematta alkuperäiseen datalähteeseen. Tämä tekee tiedon saamisen helpommaksi.

3.3.3 Metaverse

Metaverse on joukko tauluja MIIS 2003:n sisällä. Taulut sisältävät integroidun (yhdistetyn) identiteettitiedon monesta eri yhdistetystä datalähteestä. Tietystä henkilöstä kaikki identiteettitieto, joka on tallennettu moneen datalähteeseen, on synkronoitu yhdeksi merkinnäksi Metaverseen. Metaverse on suuri tietokanta, jossa tiedot yhdennetään samaksi eri datalähteistä ja periytetään mahdolliset muutokset takaisin datalähteisiin. MIIS 2003:n avulla Metaverseen voidaan kohdistaa hakuja. (Key MIIS Concepts, [viitattu 1.8.2005])

3.3.4 Rules Extensionit

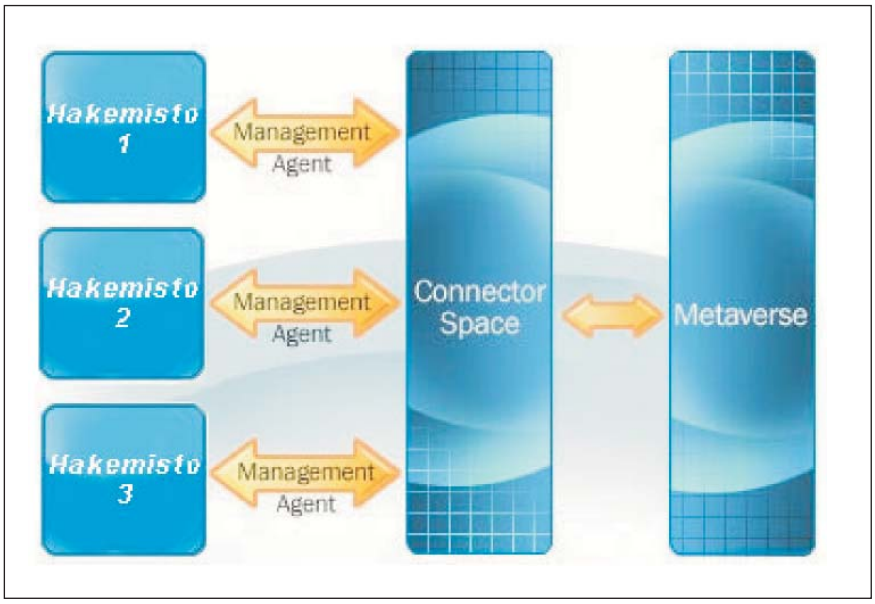
Microsoft Identity Integration Server 2003:ssa on joustava synkronointisääntömoottori, joka antaa prosessoida identiteettitiedon tietolähteen ja Metaversen välillä riippuen organisaation tarpeista. Joustavalla synkronointisääntömoottorilla tarkoitetaan ohjelmistomoottoria, johon voidaan ohjelmoida laajennuksia. Rules Extensioneita voi ohjelmoida joko Visual C# .Net 2003:lla tai Visual Basic .Net 2003:lla. Rules Extensionit toteutetaan joko Microsoft .Net Framework-luokkakirjastoina tai DLL-kirjastoina. Rules Extensionit pitää tallentaa ”extensions”-hakemistoon, joka löytyy MIIS 2003:n ohjelmakansiosta. MIIS 2003:ssa on kahdenlaisia Rules Extensioneita. Nämä ovat Metaverse Rules Extensionit ja Management Agent Rules Extensionit. (Microsoft Identity Integration Server 2003 Developer Reference, 2005)

Metaverse Rules Extensioneita käytetään tiedon virtaukseen Metaversen ja Connector Spacen välillä. Metaverse Rules Extensionit ovat vastuussa tiedon muutoksista Metaversessä, kuten attribuuttiarvojen muutoksista, linkityksien lisäyksistä tai poistamisista Metaverseobjektilta. Esimerkiksi voidaan käyttää Metaverse Rules Extensionia tekemään provisiointisääntöjä varmistamaan, että kaikki yhdistetyt hakemistot sisältävät yhteisen identiteettitiedon, joka on tallennettu Metaverseen. Provisiointi on tiedon jakamista kohdehakemistoon (Connector Spaceen). MIIS 2003 voi käyttää vain yhtä Metaverse Rules Extensionia, mutta voidaan ohjelmoida DLL-kirjasto, joka käyttää monia eri Metaverse Rules Extensioneita. (Microsoft Identity Integration Server 2003 Developer Reference, 2005)

Management Agent Rules Extensioneita käytetään tiedon virtaukseen Connector Spacen ja Metaversen välillä, kuten tiedon muutoksiin, yhdistämissääntöihin ja deprovisiointiin. Deprovisiointi on käänteinen prosessi provisioinnille, eli siinä haluttu objekti poistetaan kohdehakemistosta (Connector Spacesta). Management Agentilla voi olla vain yksi Management Agent Rules Extension. (Microsoft Identity Integration Server 2003 Developer Reference, 2005)

3.3.5 Tiedon virtaus MIIS 2003 -metahakemistossa

MIIS 2003 hallinnoi tietoa saamalla identiteettitietoja yhdistetystä datalähteestä ja tallentamalla tiedon Connector Spaceen Connector Space -objekteina. Objektit tallennetaan merkintöinä Metaverseen Metaverse-objekteina. Tämä prosessi yhdistää datalähteen objektit Metaversen objekteihin. Kuvio 4 esittää MIIS 2003:n tietovirrat.



Kuvio 4. MIIS 2003:n tietovirrat. Lähde: Microsoft Identity Integration Server 2003 Product Overview, 2004 (mukaeltu)

Suoritettaessa Management Agent MIIS 2003:ssa muutokset, jotka on tehty datalähteisiin (lisäykset, poistot ja muokkaukset) kirjoitetaan Connector Spaceen. Rules Extensionit ja filtterit suoritetaan. Tämän jälkeen koottu/saatu data kirjoitetaan Metaverseen, sillä edellytyksellä, että Management Agentilla on lupa kirjoittaa Metaverseen tietoa. Metaverse lähettää muutokset muidenkin hakemistojen Connector Spaceen, joiden kanssa tieto on määritetty synkronoitavaksi. Muiden hakemistojen Management Agentit huomaavat muutokset ja tekevät muutokset hakemistoihin vienti-komennon yhteydessä.

Tässä opinnäytetyössä ohjelmoitiin sekä Metaverse Rules Extensioneita, että Management Agent Rules Extensioneita. Metaverse Rules Extensioneita ohjelmoitiin tiedon provisiointiin eri hakemistojärjestelmiin. Management Agent Rules Extensioneita ohjelmoitiin attribuuttien luomiseen sekä Metaverseen, että kohdehakemistoihin. MIIS 2003 on oleellinen osa tätä opinnäytetyötä, koska sillä mm. luodaan uudet käyttäjäobjektit kohdehakemistoihin.

3.4 SQL Server 2000:n rooli MIIS 2003 -metahakemiston toiminnassa

Microsoft Identity Integration Server 2003 käyttää Microsoft SQL Server 2000:a tietokantanaan. MIIS 2003 ja SQL Server 2000 voidaan asentaa samalle palvelimelle

tai asennus voidaan tehdä eri palvelimille. Asennettaessa MIIS 2003 SQL Server 2000:een tehdään tietokanta nimeltä MicrosftIdentityIntegrationServer. Tietokanta sisältää seuraavat tiedot

- Metaversedatan
- Management Agentit ja suoritusprofiilit
- Management Agenttien suoritushistorian
- Connector Spacen datan
- Joinerin lokin
- Rules Extensionit.

Tietokanta ei sisällä seuraavia tietoja

- Loki-tiedostoja
- \MAData -hakemiston tietoja (Management Agent Data)
- Hallinnollisia skriptejä.

(Microsoft Identity Integration Server 2003 Help, [viitattu 3.8.2005])

Jokaisella osalla, joka on yläpuolella lueteltu, on oma tehtävänsä MIIS 2003:n toiminnassa. Suoritusprofiilit ovat profiileita eri suorituksille, esimerkiksi voidaan suoritaa tuonti- (Import) tai vientioperaatio (Export). Loki-tiedostoilla seurataan MIIS 2003:n toimintaa. Hallinnollisilla skripteillä komennetaan MIIS 2003:a ulkopuolelta, esimerkiksi voidaan suorittaa Management Agent ilman MIIS 2003:n hallintatyökalua.

MicrosoftIdentityIntegrationServer-tietokantaa ei saa suoraan muokata, koska se aiheuttaa tietokannan korruptoitumisen. Kaikki muutokset pitää tehdä MIIS 2003:n käyttöliittymien kautta. (Microsoft Identity Integration Server 2003 Help, [viitattu 3.8.2005])

4 HAKEMISTOT

Hakemisto on kuin tietokanta, mutta hakemisto auttaa sisällyttämään enemmän kuvaavaa attribuuttikantaista informaatiota. Tätä informaatiota yleensä luetaan paljon useammin kuin sitä kirjoitetaan. Hakemistot ovat nopeita antamaan vastauksia suuriin hakuihin tai hakuoperaatioihin. Hakemistoilla voi olla edellytykset replikoida informaatiota laajasti, nostaa saatavuutta, luotettavuutta ja pienentää vastausaikaa. Hakemistomääritys nykypäivän termeillä on olio-orientoitunut virtuaalinen esitys verkkoresursseille, joiden jakelu tapahtuu yli monien palvelimien hierarkisella tietokanta- tai kortistorakenteella. (JMU - Directory Services, 2005)

Tässä luvussa on tietoa metahakemistoon yhdistettävistä hakemistoista. Lisäksi on kuvattu hakemistojen rakenne ja tiedon siirtyminen hakemistojen sisällä. Seinäjoen koulutuskuntayhtymälle valittiin LDAP-hakemistotuotteeksi Active Directory Application Mode (ADAM), jonka periaatteet ja toiminta keskeisesti kuvataan tässä luvussa.

4.1 Microsoftin aktiivihakemisto (Active Directory)

Microsoftin aktiivihakemisto on hakemistopalvelu ja sen päätarkoitus on tallentaa tietoa käyttäjistä, resursseista ja verkkokokonaisuuksista, sekä tarjota tietoa kenelle tahansa tai mille tahansa, jolla on pääsy hakemistoon, riippuen pääsyoikeuksista. Tämä tieto auttaa järjestelmän pääkäyttäjiä verkon hallintaan ja peruskäyttäjiä etsimään käyttäjiä ja resursseja. (Kouti, S. & Seitsonen, M. 2004, 4)

Vaikka aktiivihakemistossa on tietokanta datalle, ei aktiivihakemisto silti ole relaatiotietokanta, kuten SQL Server. Kuten kaikki hakemistopalvelut, aktiivihakemisto on optimoitu tallentamaan suhteellisen staattista tietoa. Aktiivihakemisto tarjoaa skaalautuvaa informaatiota ja pääsyn näihin tietoihin. Aktiivihakemiston tieto esitetään objekteina, jotka voidaan tehdä jokaiselle käyttäjälle, ryhmälle ja tietokoneelle. Aktiivihakemistosta löytyy objekteja myös yhteystiedoille, tulostimille, jaetuille kansioille, ohjelmistopalveluille, konfiguraatiotiedoille ja muille resursseille. (Kouti, S. & Seitsonen, M. 2004, 4)

4.1.1 Microsoftin NOS-ympäristöjen kehitys

NOS (Network Operating System) terminä tarkoittaa verkkoympäristöä, jossa on erityyppisiä resursseja, kuten käyttäjiä, ryhmiä ja tietokonetilejä. Nämä tiedot on tallennettu keskitettyyn tallennuspaikkaan, jota kontrolloidaan. Näihin tietoihin

loppukäyttäjillä on pääsy. Yleensä NOS-ympäristö käsittää yhden tai useamman palvelimen, joka tarjoaa verkkopalveluita, kuten autentikoinnin, tilien hallinnan ja loppukäyttäjien pääsyn palveluihin. (Allen, R. & Lowe-Norris, A. 2003, 4)

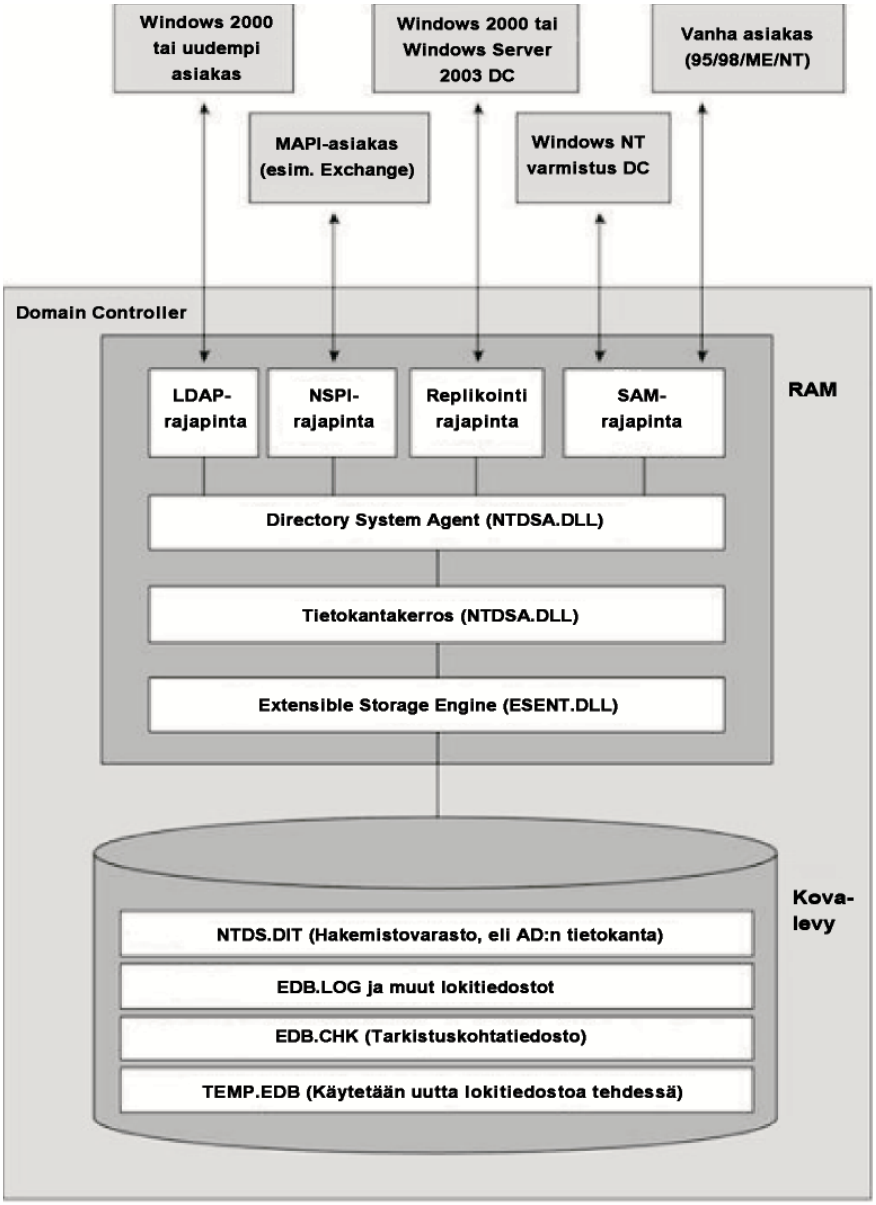
Microsoftin ensimmäinen integroitu NOS-ympäristö julkaistiin Windows NT 3.0:n mukana vuonna 1990. Windows NT 3.0 yhdisti monia eri ominaisuuksia LAN-hallintaprotokollasta ja OS/2-käyttöjärjestelmästä. Windows NT:n verkkokäyttöjärjestelmä kehittyi hitaasti seuraavat kahdeksan vuotta, kunnes aktiivihakemiston beta-versio julkaistiin vuonna 1997. (Allen, R. & Lowe-Norris, A. 2003, 4)

Windows NT:n toimialue tarjosi mahdollisuuden ryhmäresursseihin. Windows NT:n toimialue pohjautui hallintaan ja tietoturvarajoihin. NT-toimialueet ovat litteitä rakenteiltaan ja rajoitettu noin 40 000 objektiin (käyttäjät, ryhmät ja tietokoneet). Suurille organisaatioille tämä rajoitus aiheutti näkyviä rajoja toimialue-rakenteen suunnittelussa. Yleensä toimialueet ovat maantieteelliseesti rajoitettuja Domain Controllerien replikoinnin vuoksi. Toinen huomattava ongelma NT-verkkokäyttöjärjestelmän kanssa oli oikeuksien delegointi, joka yleensä oli ”kaikki tai ei mitään” -periaatteella. (Allen, R. & Lowe-Norris, A. 2003, 4)

Microsoft oli hyvin tietoinen näistä rajoitteista. Microsoftin piti uudestaan suunnitella NOS-mallinsa johonkin paljon skaalautuvampaan ja joustavampaan mallin. Tästä syystä Microsoft valitsi LDAP-pohjaisen hakemiston ratkaisuksi. (Allen, R. & Lowe-Norris, A. 2003, 4)

4.1.2 Aktiivihakemiston arkkitehtuuri

X.500-hakemistopalvelimia kutsutaan DSA:ksi (Directory System Agent) ja Microsoftin aktiivihakemisto on myös DSA. Windows 2000 Server- ja Windows Server 2003 -käyttöjärjestelmissä DSA toimii prosessista, jonka nimi on LSASS.EXE (Local Security Authority Subsystem). Tiedosto NTDSA.DLL (NT Directory System Agent) suorittaa suurimman osan toiminnoista. Jokaisella aktiivihakemiston Domain Controllerilla on aktiivihakemiston varsinainen tietokanta NTDS.DIT ja muut tiedostot kovalevyllä. Tiedostot sisältävät kaikki aktiivihakemiston tiedot. DSA komponentti tarjoaa informaatiota muille järjestelmille, jotka tarvitsevat sitä. Kuviossa 5 on kuvattu Domain Controllerin fyysinen rakenne.



Kuvio 5. Domain Controllerin fyysinen rakenne. Lähde: Kouti, S. & Seitsonen, M. 2004, 51 (mukaeltu)

Aktiivihakemiston muokatessa tietoa, se käy läpi muutaman operaation. Muokkausoperaatio on atominen, eli joko se onnistuu täydellisesti tai se ei onnistu lainkaan. Jokainen operaatio kirjoitetaan ensin lokiin ja jos operaatio onnistuu, niin se kirjoitetaan myös varsinaiseen tietokantaan (NTDS.DIT). Tietojen kirjoitus lokiin suojaa tietokantaa korruptoitumiselta ja jokainen kunnollinen tietokantatuote käyttää samanlaista tekniikkaa esimerkiksi SQL Server 2000. Yleensä hakemistotietoa haetaan ja luetaan. Lokeja tarvitaan vain silloin, kun tietoa kirjoitetaan ja muokataan. (Kouti, S. & Seitsonen, M. 2004, 51-54)

ESE (Extensible Storage Engine) hoitaa tietojen tallentamisen ja tietokantakerros hoitaa tietokannan taulut loogisella tasolla. Aktiivihakemisto käyttää vain kahta tietokannan taulua, kun normaali tietokantaohjelmisto voi käyttää kymmeniä tauluja. Objektitaulu sisältää tietoa aktiivihakemiston objekteista ja linkitystaulu toteuttaa objektien väliset yhteydet. Objektitaulua voidaan ajatella Excelin isona laskentataulukkona, jossa jokainen rivi esittää yhtä objektia ja jokainen sarake vastaa yhtä objektin attribuuttia. (Kouti, S. & Seitsonen, M. 2004, 51-54)

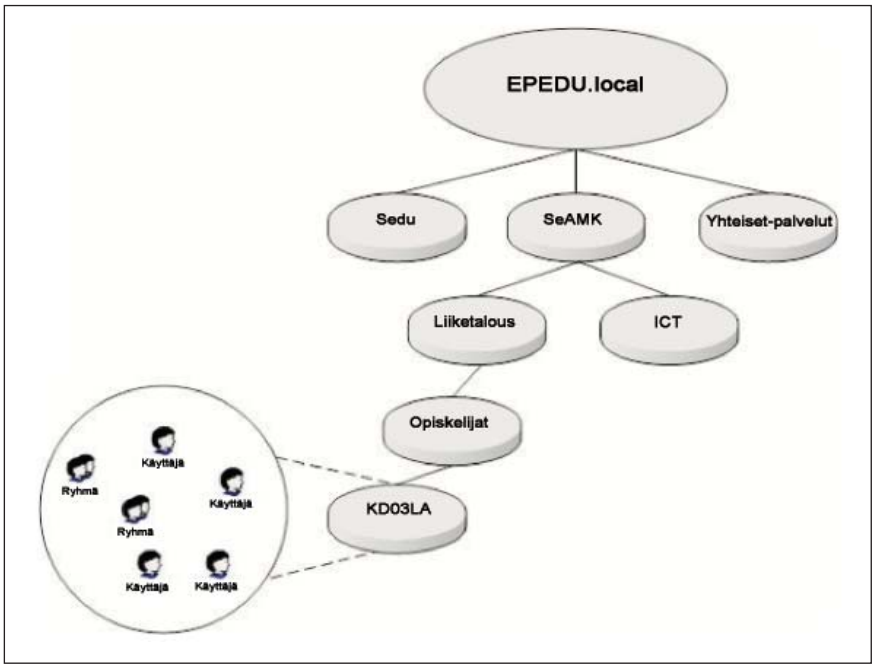
ESE käyttää ISAM-teknologiaa (Indexed Sequantial Access Method), jota ennen kutsuttiin Microsoft Jetiksi. Microsoft käyttää ISAM-taulukkohallintaa mm. Accessissa, Exchangessa, WINS:ssä (Windows Internet Name Server), FRS:ssä (File Replication Service) ja monessa muussa tuotteessaan. (Kouti, S. & Seitsonen, M. 2004, 51-54)

Tietokantakerros käsittelee tietoa käyttäen litteää tietokantamallia, rakennehierarkia puuttuu siis kokonaan. DSA (Directory System Agent) tekee hierarkisen nimiavaruuden, missä Organizational Unitit ja muut kontainerit muuntuvat puumalliseksi. DSA:ta voidaan myös kutsua hakemistokerrokseksi. (Kouti, S. & Seitsonen, M. 2004, 51-54)

Suurimpaan osaan hakemistokyselyistä käytetään LDAP-protokollaa. Normaalisti Windows 2000 tai uudemmat asiakaskäyttöjärjestelmät käyttävät LDAP-protokollaa, kuten myös muutkin LDAP-asiakasohjelmistot. (Kouti, S. & Seitsonen, M. 2004, 51-54)

4.1.3 Aktiivihakemiston objektihierarkia

Data aktiivihakemistossa tallennetaan hierarkiseen muodostelmaan, kuten normaalissa tiedostojärjestelmässäkin. Jokainen merkintä on objekti. Rakennekerroksella on kahdentyyppisiä objekteja kontainerit (säiliö) ja kontainerittomattomat, jotka tunnetaan myös nimellä lehtisolmut. Juurihakemistosta (root) haarautuu yksi tai useampi kontaineri. Jokainen kontaineri voi sisältää lehtisolmun tai muita kontainereita. Lehtisolmu, kuten nimi viittaakin, ei voi sisältää muita objekteja. Kuvio 6 kuvaa esimerkkiä aktiivihakemistosta.



Kuvio 6. Esimerkki aktiivihakemistosta.

Kuvio 6 esittää aktiivihakemiston vanhempi-lapsi suhdetta. Puun juurella (root) on kolme lasta Sedu, SeAMK ja Yhteiset-palvelut. Kaikki nämä ovat kontainereita muille objekteille. SeAMK:lla on kaksi lasta Liiketalous ja ICT. Liiketalous-kontainerin lapsiobjektit ovat näkyvissä. Liiketalous-kontaineri sisältää Opiskelijat-kontainerin ja se taas sisältää KD03LA-kontainerin. KD03LA-kontaineri sisältää käyttäjiä ja ryhmiä. Jokaisen lapsisolmun (esim. käyttäjä) vanhempi on KD03LA-kontaineri. Kyseinen kuva tunnetaan myös aktiivihakemistossa toimialueena.

4.1.4 Objektien tunnistaminen aktiivihakemistossa

Aktiivihakemiston sisältäessä tuhansia objekteja, jokaisella objektilla on oltava oma tunniste ja sijoituspaikkansa. Jokaiselle objektille määritetään GUID (Globally Unique Identifier), kun objekti luodaan järjestelmään. Microsoft takaa tämän 128-bitin numerosarjan olevan yksikäsitteinen. Poistettaessa objekti aktiivihakemistosta myös objektin GUID poistuu. Objektia muokattaessa tai siirrettäessä hakemistopuussa säilyy sama GUID kaiken aikaa.

GUID:n ollessa yksiselitteinen ja joustava, sitä ei ole kovin helppo muistaa. GUID ei perustu hakemistohierarkiaan. Toinen tapa viitata objekteihin on aktiivihakemistossa hierarkkiset polut eli ADsPathit ja se on yleisimmin käytetty.

4.1.5 Aktiivihakemiston hierarkiset polut

Aktiivihakemiston hierarkiset polut tunnetaan nimellä ADsPathit. ADsPatheja voidaan käyttää objektien viittaamiseen. Microsoft käyttää ADsPatheja suurimmissa hakemistoissa, kuten esimerkiksi aktiivihakemistossa, Windows NT:ssä ja monessa muussa järjestelmässään.

Aktiivihakemiston hierarkiset polut objekteille esitetään LDAP-muodossa käyttäen syntakseja ja sääntöjä, jotka ovat määritetty LDAP-standardeissa. Kuviossa 6 olevan esimerkin juuren voidaan viitata seuraavasti:

LDAP://DC=EPEDU,DC=local

Polku alkaa LDAP:in ohjelmallisella tunnistella (progID) ja sitä seuraa kaksoispiste (:) ja kaksi kappaletta kenoviivoja (//).

Viitattaessa tiettyyn toimialueen juureen, jokainen osa pitää erottaa pilkulla (,) ja jokaisen osan alkuliite on DC. Esimerkiksi, jos toimialueen nimi on EsimDomain. Organisaatio.org, tällöin polku näyttäisi seuraavalta:

LDAP://DC=EsimDomain,DC=Organisaatio,DC=org

DC tulee sanoista Domain Component ja sitä käytetään määrittämään toimialueet tai eri ohjelmisto-osiot. (Allen, R. & Lowe-Norris, A. 2003, 14-16)

Distinguished Name (DN) on nimi, jota käytetään objektien yksikäsitteiseen tunnistamiseen hakemistopuussa. Relative Distinguished Name (RDN) on nimi, jota käytetään objektien tunnistamiseen vanhempi-objektin sisäpuolella. Esimerkiksi oletus Administrator-tilin ADsPath, joka sijaitsee Users-kontainerin sisällä Organisaatio.org toimialueessa, on seuraava:

LDAP://CN=Administrator,CN=Users,DC=Organisaatio,DC=Org

Saman käyttäjän DN on seuraava (progID puuttuu):

CN=Administrator,CN=Users,DC=Organisaatio,DC=Org

Saman käyttäjän RDN on seuraava:

CN=Administrator

Nämä polut kootaan nimistä ja alkuliitteistä ja erotetaan yhtä kuin (=) -merkillä. Yksi yleisimmistä alkuliitteistä on OU (Organizational Unit). Esimerkki OU:sta:

CN=Matti Meikäläinen,OU=Osasto 1,DC=Organisaatio,DC=org

Kaikki RDN:t, DN:t ja ADsPathit käyttävät alkuliitteitä, jotka ilmaisevat objektin luokan, johon viitataan. Mikä tahansa objekti, jolla ei ole kirjainkoodia käyttää oletusta, joka on CN (Common Name). Taulukkoon 1 on koottu yleisimmin käytetyt alkuliitteet, joita yleisimmät hakemistopalvelimet käyttävät. Täydellinen dokumentti RFC 2253:n määrityksistä on saatavissa IETF:ltä (The Internet Engineering Task Force) (IETF, [viitattu 20.10.2005]).

Taulukko 1. Yleisimmin käytetyt alkuliitteet hakemistoissa. Lähde: Allen, R. & Lowe-Norris, A. 2003, 14-16

Avain	**Attribuutti**
CN	Common Name
L	Locality Name
ST	State Name
O	Organization Name
OU	Organizational Unit Name
C	Country Name
STREET	Street Address
DC	Domain Component
UID	Userid

Microsoft Exchange 5.5 käyttää O-alkuliitteitä, aktiivihakemisto käyttää ainoastaan DC:tä, CN:ää ja OU:ta, joista CN:ää käytetään useimmissa tapauksissa. (Allen, R. & Lowe-Norris, A. 2003, 14-16)

Tässä opinnäytetyössä hierarkisia polkuja hyödynnetään mm. OU-provisioijan tehdessä OU-haaroja kohdehakemistoihin ja Metaverse Rules Extensioneiden objektien jakamisessa eri OU:hin.

4.2 Active Directory Application Mode (ADAM)

Active Directory Application Mode on Microsoftin kehittämä LDAP-hakemisto-tuote. ADAM on suunniteltu etenkin hakemisto-ohjelmiston tarpeisiin. ADAM toimii niin sanottuna ei-käyttöjärjestelmäpalveluna ja näin ollen sitä ei tarvitse asentaa Domain Controllerille. Toiminta ei-käyttöjärjestelmäpalveluna tarkoittaa, että ADAM:in ilmentymiä voi samanaikaisesti olla monta samalla palvelimella. Jokainen ilmentymä voidaan konfiguroida toisistaan riippumatta omanlaisikseen.

Active Directory Application Mode edustaa uutta hakemistopalveluteknologiaa, joka tarjoaa joustavuutta ja auttaa organisaatioita välttämään kasvavia infrastruktuurin hoitokustannuksia. (Smit, R., [viitattu 20.8.2005])

4.2.1 Johdanto ADAM:iin

Active Directory Application Mode (ADAM) on saanut alkunsa 1990-luvun puolivälissä LDAP-pohjaisten sovellusten myötä. Organisaatiot olivat menestyksekkäästi kehittäneet hakemisto-ohjelmistoja liiketoiminnan avainongelmiin. Ohjelmistot olivat lähinnä käyttäjän todentamiseen liittyviä sovelluksia, kuten valkoisten/keltaisten sivujen todentamispalveluita ja tietovirtaohjelmistoja liiketoiminnan avuksi (line-of-business).

Kehityksen tuloksesta monessa organisaatiossa on nykyisin ainakin yksi hakemistopalvelu, jolla käyttäjät todennetaan. Muita hakemistopalveluita organisaatioissa ovat mm.

- etäyhteydet organisaation omaan verkkoon (VPN)
- valkoiset sivut (hakemistopalveluna)
- extranet tai Web SSO.

Organisaatioille on hyvin yleistä kehittää monia eri hakemistopalveluita ja käyttää hakemistoja, jotka pohjautuvat eri hakemistoteknologioihin. Esimerkiksi organisaatiolla voi olla aktiivihakemisto, jolla käyttäjät todennetaan verkkoon, kun taas extranet-todennus hoidetaan esimerkiksi X.500-hakemistojärjestelmästä ja valkoiset sivut taas käyttävät eri hakemistojärjestelmää. (Introduction to Windows Server 2003 Active Directory Application Mode 2005.)

Jos jokainen näistä järjestelmistä pohjautuu LDAP-hakemistopalveluun, niin miksi organisaatiot eivät ole onnistuneet standardisoimaan yhtä hakemistoteknologiaa? Esteitä ovat olleet mm. seuraavat:

- hakemistojen yhteensopimattomuus
  - Useimmat hakemistopalvelut eivät vain toimi toistensa kanssa. Historiallinen esimerkki on alkuperäinen X.500-hakemisto, joka ei tukenut LDAP-protokollaa. Nykyäänkin löytyy hakemistotuotteita, jotka eivät tue LDAP-protokollaa tai muita laajalti käytettyjä protokollia.
- vaihtoehtojen vähäisyys
  - Ohjelmistotekijät tekevät hakemistotuotteita, jotka kattavat vain osan hakemistopalveluista, joita nykyään käytetään. Näin ohjelmistotekijöiden asiakkaat voidaan pakottaa (ohjelmistotukeen riippuvista syistä) käyttämään hakemistopalvelua, jota ei vielä löydy asiakkaan organisaatiosta.

- koordinaation puute
  - Joissakin tapauksissa on organisaatioyksiköitä, jotka on eristetty toisistaan. Tällöin organisaatioissa asennetaan eri liiketoimintaratkaisuja, jolloin voi olla käytössä monia eri hakemistoratkaisuja.
- tietoturvan yhteensopimattomuus
  - Liiketoimintaratkaisut harvoin antavat oikeuden käyttää identiteettitietoja, jotka on tallennettu tiettyyn hakemistoon. Identiteettitiedot eivät ole yhteensopivia kaikkien ratkaisujen kanssa. Näin kehitetään uusia hakemistoratkaisuja, jotka toimivat tietoturvakriteerien sisällä.

Monessa organisaatiossa on piilokuluja, jotka johtuvat monen eri hakemistoteknologian käytöstä. Nämä kulut ovat mm. seuraavia:

- lisääntyvä tietoturvariski
  - Työntekijät, partnerit, kontaktit tai asiakkaat vaihtavat suhdettaan organisaatioon usein. Tällöin on ensiarvoista, että heidän oikeutensa eri hakemistoihin muutetaan suhteen mukaan. Esimerkiksi, kun työntekijä eroaa organisaatiosta ja tulee organisaation asiakkaaksi. Tällöin hänen ei pidä päästä samoihin resursseihin käsiksi kuin ennen.
- kallis hinta
  - Jokainen liiketoimintaratkaisu, joka pohjautuu eri hakemistoteknologiaan, organisaatiossa vaatii seuraavia ominaisuuksia
    - kyseiseen hakemistoteknologiaan koulutetun henkilökunnan
    - käyttö- ja hallintotoiminnot
    - lisälisenssien ja tukisopimusten ylläpidon.
- lisenssien hinnan nousu.
  - Jotkut hakemistot on lisensoitu sen mukaan, montako objektia hakemistossa on.
- liiketoimintaprosessien integroinnin puute
  - Hakemistotieto voi vaihdella. Käyttäjät siirtyvät ryhmästä toiseen, vaihtavat toimiston paikkaa, puhelinnumeroa, vaihtavat sukunimeä ja arvonimikettään. Tällöin tieto pitää päivittää eri hakemistoihin. Ilman automatisoitua prosessia tieto vanhentuu eri hakemistoissa.

Organisaatiot tarvitsevat hakemiston, jota he voivat kehittää omaan NOS-infrastruktuuriinsa. Active Directory Application Mode mahdollistaa tällaisen hakemistoinfrastruktuurin ilman kallista koulutusta, lisälisensointeja tai käyttökustannuksia, jotka voivat johtua lisäkomponenttiasennuksista hakemistoteknologiaan. Hakemistoihin lisättäessä ominaisuuksia, lisäominaisuudet maksavat yleensä paljon. (Introduction to Windows Server 2003 Active Directory Application Mode 2005.)

Active Directory Application Mode on uusi muoto Microsoftin aktiivihakemistosta, joka vastaa hakemisto-ohjelmistojen tarpeisiin. ADAM toimii niin sanottuna ei-käyttöjärjestelmäpalveluna ja eikä vaadi Domain Controlleria. (Introduction to Windows Server 2003 Active Directory Application Mode 2005.)

4.2.2 ADAM:in toiminta

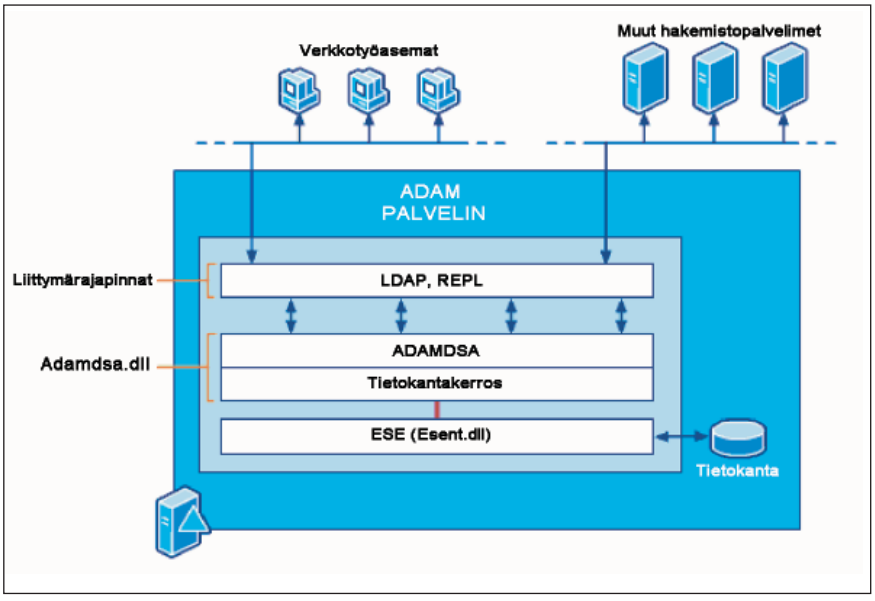
Active Directory Application Mode on LDAP-hakemistopalvelu, joka toimii käyttäjälähtöisenä palveluna. ADAM:ia voidaan käyttää Windows Server 2003 -perheen käyttöjärjestelmissä ja Windows XP SP1 -käyttöjärjestelmässä. ADAM:ia ei tarvitse asentaa Domain Controllerille. Monia ADAM:in ilmentymiä voidaan asentaa samalle tietokoneelle ja käyttää samaan aikaan.

4.2.2.1 ADAM:in arkkitehtuuri

Active Directory Application Moden arkkitehtuuri sisältää muutamia komponentteja, jotka toimivat samaan aikaan tarjotakseen hakemistopalvelun. Komponentit ovat seuraavat

- liittymärajapinnat (LDAP, replikointi ja hallinta rajapinnat (REPL))
- Directory System Agent (DSA)
- tietokantakerros
- Extensible Storage Engine (ESE)
- hakemistotietokanta.

Kuviossa 7 on kuvattu ADAM:in arkkitehtuuri.



Kuvio 7. ADAM:in arkkitehtuuri. Lähde: Active Directory Application Mode Technical Reference (Draft), 2004 (mukaeltu)

Liittymärajapinnat (LDAP ja REPL)

- Tietovaraston käyttöliittymät tarjoavat keinon, jolla verkkotyöasemat ja muut hakemistopalvelimet voivat kommunikoida tietovaraston kanssa.

DSA

- Directory System Agent (Adamdsa.dll), joka toimii jokaisella Domain Controllerilla, tarjoaa käyttöliittymän, jolla määritetään, millä hakemistoasiakkailla ja ADAM:in ilmentymillä on pääsy hakemistotietokantaan. DSA pakottaa hakemistosemantiikkaan, pitää yllä skeemaa, takaa objektien identiteettisyyden ja pakottaa tietotyypit attribuuteille.

Tietokantakerros

- Tietokantakerros on API (Application Programming Interface), joka löytyy Adamdsa.dll tiedostossa ja tarjoaa rajapinnat hakemiston ja ohjelmiston välille. Hakemisto-ohjelmistoilla ei voida ottaa suoraa yhteyttä tietokantaan, koska tietokantakerros suojaa sitä suoralta kanssakäymiseltä. Ohjelmistoista ei koskaan tehdä suoria kutsuja tietokantaan, vaan kutsut menevät tietokantakerroksen läpi. Hakemistotietokannan ollessa litteä, ilman hierarkista nimiavaruutta, tietokantakerros tarjoaa tietokannan objektihierarkian abstraktina.

ESE

- Extensible Storage Engine (Esent.dll) hallinnoi taulukon tietueita (yksi sarake tai enemmän). ESE valmistaa hakemistotietokannan.

Hakemistotietokanta

- Tietovarasto tallentaa tietoa yhteen tietokantatiedostoon. Tiedoston nimi on Adamntds.dit. Tietovarasto käyttää lokeja, joihin se kirjoittaa väliaikaisia merkintöjä sitoutumattomista tapahtumista.

4.2.2.2 ADAM:in fyysinen rakenne

Active Directory Application Moden fyysinen rakenne rakentuu seuraavista komponenteista

- tietokoneista, joissa ADAM on asennettuna (ADAM-palvelimet)
- ADAM:in ilmentymistä
- ADAM:in konfiguraatioista ja
- sijainneista.

Tietokonetta, jossa ADAM on asennettuna, sanotaan ADAM-palvelimeksi. Joka kerta, kun ADAM asennetaan, luodaan uusi yksikäsitteinen ADAM-ilmentymä.

Jokainen ADAM-ilmentymä koostuu seuraavista osista

- ohjelmatiedostoista
- datatiedostoista
- rekisteriavaimista
- palvelun nimestä
- palvelun päätilistä
- tapahtumalokista
- nimi:portti yhdistelmästä.

Asennuksen jälkeen, ADAM-ilmentymän ohjelmatiedostot ja työkalut löytyvät hakemistosta %windir%\ADAM. Asennettaessa useita ADAM-ilmentymiä samalle tietokoneelle, ADAM:in työkalut ja ohjelmatiedostot ovat jaettuna kaikille ilmentymille. Päivitettäessä joku ADAM-ilmentymistä, joka on samalla tietokoneella, päivitys kohdistuu tällöin myös kaikkiin muihinkin ilmentymiin. (Active Directory Application Mode Technical Reference (Draft), 2004)

Active Directory Application Moden datatiedostot asennetaan hakemistoon %programfiles%\Microsoft ADAM\ilmentymän nimi\Data, jossa ilmentymän nimi on asennuksen aikana annetun ilmentymän nimi.

Asennettaessa monta ADAM-ilmentymää samalle tietokoneelle, jokaisella ilmentymällä on oma datatiedosto. Esimerkiksi asennettaessa kaksi ADAM-ilmentymää, joiden nimet ovat ilmentyma1 ja ilmentyma2, hakemistot tällöin näyttävät seuraavanlaisilta

- Program Files\Microsoft ADAM\ilmentyma1\Data
- Program Files\Microsoft ADAM\ilmentyma2\Data.

Kaikki ADAM:in rekisteriavaimet tallennetaan seuraavaan sijaintiin rekisterissä:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\
Services\ADAM_ilmentymannimi.

Asennuksen aikana annetaan nimi ADAM-ilmentymälle, jota käytetään hakemistorakenteen ja rekisteriavainten luomiseen. Nimeä, joka annetaan ADAM-ilmentymälle, käytetään myös palvelun ja näyttönimen nimeämiseen.

Nimi, joka annetaan ADAM-ilmentymälle, tulee täyttää seuraavat vaatimuksen:

- Nimen pitää olla yksikäsitteinen eli, kun samalla tietokoneella on useita ADAM-ilmentymiä, nimi ei saa olla sama minkään ilmentymän kanssa.
- Nimessä saa olla korkeintaan 44 merkkiä.
- Merkit voivat olla normaaleita aakkosia ja numeroita eli ei skandinaavisia merkkejä.
- Nimeä ”ntds” ei saa käyttää.

Palvelun näyttönimi ilmestyy lisää tai poista sovellus -ikkunaan ja Microsoft Management Consoleen. Palvelun nimen muokkaamiseen on käytössä työkalu nimeltä sc. Kyseistä työkalua voidaan käyttää milloin tahansa.

ADAM-ilmentymät toimivat käyttäjäpalveluna. ADAM-ilmentymille määritetään asennuksen aikana päätili, jolla on täydet oikeudet tietokantaan. Tilin voi valita Windows-toimialueesta tai -työryhmästä, johon ADAM asennetaan. (Active Directory Application Mode Technical Reference (Draft), 2004)

Active Directory Application Moden asennusvelho auttaa valitsemaan palvelun päätilin, joka on ympäristössä. Taulukkossa 2 on esitelty tilit, jotka ovat hyväksyttyjä ympäristössä.

Taulukko 2. Tilit, jotka ovat hyväksyttyjä ADAM-ympäristössä. Lähde: Active Directory Application Mode Technical Reference (Draft), 2004 (mukaeltu)

Ympäristö	Palvelun päätili	
	Ensimmäinen ADAM-ilmentymä	**Replikoitu ADAM-ilmentymä**
Työryhmä, joka ei ole toimialueessa tai metsässä.	Verkkopalvelu.	ADAM-ilmentymien replikointi ei sallittua.
	Työaseman käyttäjä.	Työaseman käyttäjä.
Windows 2000 Server toimialue tai metsä tai Windows Server 2003 toimialue tai metsä.	Verkkopalvelu.	Verkkopalvelun tai toimialueen käyttäjä.
	Työaseman käyttäjä.	Työaseman käyttäjä.
	Toimialueen käyttäjä.	Toimialueen käyttäjä tai verkkopalvelun käyttäjä.
Windows NT 4.0 toimialueet.	Työaseman käyttäjä.	Työaseman käyttäjä.
	Toimialueen käyttäjä.	Toimialueen käyttäjä.

Palvelun päätiliä ei tulisi muuttaa kuin dsmgmt-työkalulla.

ADAM-ilmentymät kirjoittavat tapahtumat tapahtumalokiin. Näitä tapahtumia voi selata tapahtumienvalvonta-työkalulla (Event Viewer).

Hakemisto-ohjelmistojen, jotka ovat yhteydessä ADAM-ilmentymiin, täytyy määrittää NetBIOS-nimi (Network Basic Input/Output System), DNS-nimi (Domain Name Service) tai IP-osoite, jossa ADAM-ilmentymä sijaitsee. Ohjelmistojen täytyy määrittää LDAP- tai SSL-viestintäportti (Secure Socket Layer), jota ADAM-ilmentymä käyttää. Esimerkiksi, kun hakemisto-ohjelmisto haluaa

ottaa yhteyden ADAM-ilmentymään, joka toimii tietokoneella, jonka DNS-nimi on ADAMPalvelin.Esimerkki.org, NetBios-nimi on ADAMPalvelin ja IP-osoite on 192.168.111.101. Esimerkkipalvelin käyttää porttia 50000 LDAP-yhteyksille. Tällöin hakemisto-ohjelmistot voivat käyttää jotain seuraavista osoitteista

- ADAMPalvelin.Esimerkki.org:50000
- ADAMPalvelin:50000
- 192.168.111.101:50000.

(Active Directory Application Mode Technical Reference (Draft), 2004)

4.2.3 ADAM:in hyödyt

Active Directory Application Mode on laajennettu muoto Microsoftin aktiivihakemistosta. ADAM on hakemistopalvelu, joka antaa nopean toteutusmahdollisuuden hakemisto-ohjelmistoille. Active Directory Application Mode tarjoaa seuraavissa luvuissa esitellyt hyödyt.

4.2.3.1 Runsas ja laaja tallennusvarasto

ADAM tukee joustavaa ja laajennettavaa skeemaa. Erilaisia työkaluja hyödyntämällä voi helposti muokata skeemaa, kuten ldifde, ADAM schema ja ADAM ADSI Edit, jotka ovat samanlaisia työkaluja, kuin aktiivihakemistossakin. Jokaisella ADAM-ilmentymällä, joka on asennettu samalle tietokoneelle, on oma skeemansa.

Yksi ADAM ilmentymä voi isännöidä monia eri dataosioita, näin ollen voidaan määrittää muisti-, jakelu- ja replikointilaajuus datalle eri osioissa. ADAM:in tallennusvarasto tarjoaa joustavan nimiavaruuden, tukien molempia DNS-tyylisiä ja X.500-tyylisiä DN:iä (Distinguished Name). DN identifioi hakemistossa olevan objektin tarkasti ja yksikäsitteisesti. DN on koottu objektin tarkasta polusta esimerkiksi CN=Meikäläinen Matti, OU=KD03LA, OU=Opiskelijat, OU=Liiketalous, OU=SeAMK, DC=EPEDU, DC=local.

Tämä saa aikaan nopean hakemiston kehityksen, joka vaatii vähemmän suunnittelua skeemalle ja nimeämistavalle. (Introduction to Windows Server 2003 Active Directory Application Mode, 2005)

4.2.3.2 Replikointi

Active Directory Application Mode käyttää samaa monireplikointimallia kuin aktiivihakemisto, mikä varmistaa, että replikoitua tietoa voidaan muokata missä tahansa ADAM-ilmentymässä, joka osallistuu replikointijoukkoon. ADAM:in replikointi käyttää samaa sijaintimallia kuin aktiivihakemisto ja se tarjoaa ominaisuuksia, kuten

pakatun replikointimaaston, joka voidaan ajastaa. Replikointia voidaan hallinnoida tutuilla replikointivälineillä, kuten repadmin-ohjelmalla.

Aktiivihakemistossa tiedon replikointi on jatkuvaa aktiivihakemistojen kesken. Aktiivihakemistossa yleensä määritetään tietty aika, jolloin tiedot replikoituvat kaikkien aktiivihakemistojen kesken. ADAM:ssa on mahdollisuus päättää milloin ja mihin ADAM:in ilmentymään replikointi tapahtuu.

ADAM voi replikoida tiedon moneen eri ilmentymään. ADAM-ilmentymäreplikoinnin voi suorittaa palvelimilla, jotka ovat aktiivihakemiston toimialueen jäseniä. ADAM-ilmentymän replikoinnin voi myös suorittaa palvelimilta, jotka ovat eri toimialueissa, metsissä tai eri työryhmien jäseniä. (Introduction to Windows Server 2003 Active Directory Application Mode, 2005)

4.2.3.3 Asennus ja poisto

Active Directory Application Moden asennus käyttää tuttua Windows-pohjaista asennusohjelmaa. Käskyjä asennuksen aikana tarvitaan vähän, joihin voidaan myös tehdä erillinen skripti. Skriptin tekeminen on suositeltavaa muokattuihin ja hiljaisiin asennuksiin. Tämä helpottaa ADAM:in uudelleenasennusta. Active Directory Application Moden asennusohjelmalla voidaan tehdä uusi ilmentymä tai ottaa kopio (replika) olemassa olevasta ilmentymästä.

Active Directory Application Mode tarjoaa puhtaan ohjelmistopoistoprosessin, joka poistaa seuraavat osat:

- Ilmentymät, jotka valitaan.
- Kaikki tiedostot, jotka sisältävät konfiguraatiodataa.
- Yhteenkuuluvat osiot.

ADAM:in asennus ja poisto säästävät aikaa pääkäyttäjiltä ja helpottavat ADAM:in siirtoa testiympäristöstä varsinaiseen käyttöympäristöön.

4.2.3.4 Tuki monelle ilmentymälle

Usea ADAM-ilmentymä voi toimia samalla palvelimella yhtä aikaa. Jokainen ADAM-ilmentymä voidaan konfiguroida itsenäiseksi ja eristää muilta ilmentymiltä, jotka toimivat samalla palvelimella. ADAM-ilmentymät tunnistetaan yksikäsitteisellä nimellä ja portilla, jotka voidaan asennuksen aikana määrittää.

Moni ilmentymätuki ADAM:ssa tarjoaa merkittävän edun organisaatiolle, kuten palvelinten yhdistämisen ja hakemisto-ohjelmistojen kehityksen. Pienimmissä

organisaatioissa, voidaan käyttää moni-ilmentymätukea hakemisto-ohjelmistokehitykseen siten, että jokaiselle ohjelmalle tehdään oma ilmentymä.

4.2.3.5 Varmuuskopiointi ja palautus

Active Directory Application Modessa on integroidut varmuuskopiointi- ja palautustoiminnot, jotka löytyvät Windows-käyttöjärjestelmistä. Jokaisen ADAM-ilmentymän varmuuskopiointi voidaan tehdä joko ajastettuna tai online prosessina, jos on pääsy ADAM:in kriittiseen dataan. ADAM:in palautus toimii varmuuskopiointityökalulla joko online- tai offline-tilassa. (Introduction to Windows Server 2003 Active Directory Application Mode, 2005)

4.2.3.6 ADAM:in tukemat työkaluohjelmistot

Active Directory Application Mode on uusi muoto aktiivihakemistosta, siksi ADAM:in hallinnointi on hyvin samanlaista kuin aktiivihakemistonkin. ADAM käyttää samoja tuttuja työkaluohjelmistoja kuin aktiivihakemisto. ADAM:in työkaluohjelmistot asennetaan ADAM:in asennuksen aikana. ADAM:in hallinointiin voidaan käyttää seuraavia työkaluja:

- Ldp-ohjelmalla voidaan tehdä LDAP-operaatioita ADAM:iin. Ldp on osa tukityökaluja Windows 2000 Server- ja Windows Server 2003 -perheissä. Ldp käyttää graafista käyttöliittymää.

- ADAM ADSI Edit perustuu tuttuun ADSI Edit työkaluohjelmaan. ADAM ADSI Edittiä voidaan näyttää, muokata ja luoda ACL:iä (Access Control List) kaikille objekteille hakemistosta. ADAM ADSI Editillä voidaan myös näyttää ja muokata skeemaa ja konfiguraatiotietoja.

- Tutut työkalut, kuten PerfMon-suorituskykykonsoli soveltuu ADAM:in valvontaan. Tietoa voidaan kerätä jokaisesta ADAM-ilmentymästä laskureilla ja jäljityslokeilla. Näyttöresursseja voidaan muokata Microsoft Management Consolella (MMC).

- Komentorivityökaluja, kuten dsmgmt ja dsdbutil, voidaan käyttää tietokannan ylläpitoon, hallintaan, kontrollointiin ja yhden käskyn operaatioihin. Näillä komentorivityökaluilla voidaan poistaa kelvoton metatieto ja luoda hakemisto-osioita.

Active Directory Application Mode säästää rahaa ja opetusaikaa, koska ADAM:ssa voidaan käyttää samoja tuttuja työkaluja kuin aktiivihakemistossa. Työkalut, jotka ovat oleellisia tässä opinnäytetyössä ADAM:in asennuksessa ja konfiguroinnissa, ovat ldifde ja ADAM ADSI Edit. Ldifde-työkalulla voidaan luoda, muokata ja

poistaa hakemisto-objekteja. Ldifde-työkalulla voidaan laajentaa skeemaa. ADAM ADSI Edit -työkalulla voidaan selata, luoda, poistaa ja muokata hakemisto-objekteja ADAM:lta. (Introduction to Windows Server 2003 Active Directory Application Mode, 2005)

4.2.3.7 Tietoturva

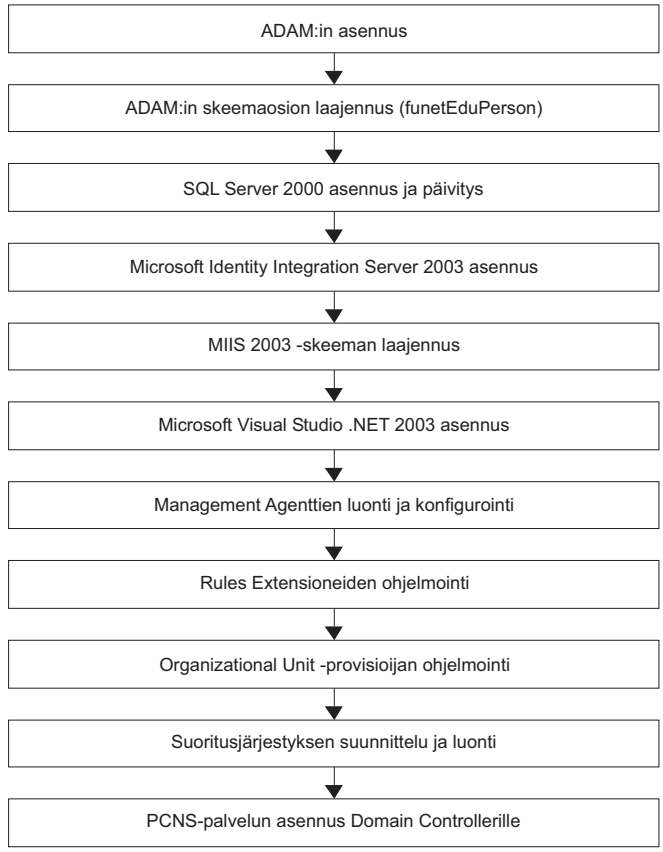
Active Directory Application Mode käyttää Windows-käyttöjärjestelmän tietoturvamallia. Pääsy objekteihin voidaan sallia ADAM:in sisältä, kuten aktiivihakemistossakin. Active Directory Application Mode tukee LDAP-autentikointia. ADAM:iin voidaan tehdä käyttäjätilejä ohjelmistojen luottaessa hakemistopalvelun autentikointiin.

Hakemisto-objektien käyttö ADAM:in kautta perustuu Windowsin ACL-malliin. Tätä pääsymekanismia voidaan käyttää pääsyoikeuksiin jokaiseen ADAM:in objektiin ja ilmentymiin. Tämä pääsymekanismi perustuu tietoturvakuvaajiin, jotka löytyvät Windowsin tietoturvainfrastruktuurista. Ohjelmistot voivat lisätä tätä pääsymekanismia käyttäen ohjelmiston omaa Frameworkia käyttöoikeuksiin samalla, kun ohjelmistot käyttävät hakemistopalvelun tarjoamaa autentikointia. (Introduction to Windows Server 2003 Active Directory Application Mode, 2005)

ADAM valittiin Seinäjoen koulutuskuntayhtymän LDAP-hakemistotuotteeksi. ADAM:in skeemaosioon päätettiin lisätä funetEduPerson-skeema. FunetEduPerson-skeemaa käytetään ADAM:ssa käyttäjäobjektien tyyppinä. FunetEduPerson-skeema muokattiin X.500-skeemamuotoon.

5 METAHAKEMISTON ASENNUS, KONFIGUROINTI JA OHJELMOINTI

Metahakemiston asennus, konfigurointi ja ohjelmointi etenee yhdentoista eri vaiheen kautta. Prosessissa viedään läpi metahakemiston ja LDAP-hakemiston asennus, konfigurointi ja ohjelmointi Seinäjoen koulutuskuntayhtymän tarvetta varten. Testiympäristön rakentaminen tämänkaltaista järjestelmää käyttöönotettaessa on suositeltavaa. Ilman testiympäristön luomista järjestelmää on melko mahdotonta ottaa käyttöön, koska testaamisella saadaan järjestelmästä pois suurimmat virheet. Kuvio 8 esittää asennus-, konfigurointi- ja ohjelmointiprosessien etenemisvaiheet.



Kuvio 8. Asennus-, konfigurointi- ja ohjelmointiprosessien eteneminen.

Asennus, konfigurointi ja ohjelmointi alkaa ADAM:in asennuksella halutulle palvelimelle. FunetEduPerson-skeemaa muokataan X.500-skeemamuotoon, ennen kuin sillä laajennetaan ADAM:in skeemaosiota. Tämän jälkeen asennetaan MIIS 2003 -palvelimelle SQL Server 2000 ja päivitetään siihen Service Pack 3. SQL Server 2000 toimii MIIS 2003:n tietokantana. Seuraavaksi asennetaan MIIS 2003 ja laajennetaan sen skeemaa tarvittavilla attribuuteilla. Tämän jälkeen asennetaan Visual Studio .Net 2003 -ohjelmistokehitystyökalu MIIS 2003 -palvelimelle. Rules

Extensioneiden ohjelmointiin tarvitaan ohjelmistokehitystyökalu. MIIS 2003 osaa integroitua Visual Studio .NET 2003:n kanssa. Tämä ominaisuus on hyödyllinen Rules Extensioneita ohjelmoitaessa. Tämän jälkeen luodaan ja konfiguroidaan Management Agentit Winhalle, aktiivihakemistolle ja ADAM:lle. Management Agenttien luonnin ja konfiguroinnin jälkeen ohjelmoidaan tarvittavat Rules Extensionit Management Agenteille ja Metaverselle. Koska MIIS 2003 ei pysty luomaan Organizational Unitteja, tätä ominaisuutta varten tarvitaan sovellus, joka tekee Organizational Unitit tarpeen vaatiessa. Organizational Unittien tekoa varten ohjelmoidaan komentorivisovellus, joka luo Organizational Unitit MIIS 2003:n virhelokista. Tämän jälkeen suunnitellaan ja luodaan Management Agenttien suoritusjärjestys. Salasanojen replikointia varten MIIS 2003:n ja Domain Controllereiden välillä tarvitaan palvelu, joka replikoi salasanat. Password Change Notification Service (PCNS) on palvelu, joka on hoitaa salasanojen synkronoinnin Domain Controllerin ja MIIS 2003:n välillä. PCNS toimitetaan MIIS 2003:n mukana. Kaiken tämän jälkeen suoritetaan testaukset. Testauksessa havaitaan mahdolliset virheet ja ne korjataan, sekä testataan uudelleen.

Testiympäristöä varten tarvitaan kahta eri palvelinta. Toinen palvelin toimii hakemistopalvelimena ja toinen metahakemistopalvelimena. Hakemistopalvelimella toimii kaksi eri hakemistoa LDAP-hakemisto ADAM ja aktiivihakemisto. Metahakemistopalvelimella toimii MIIS 2003, SQL Server 2000 ja ohjelmistokehittimenä Visual Studio .Net 2003. Tämä jako on loogisin, koska hakemistopalvelimet on hyvä pitää erillään varsinaisesta metahakemistosta.

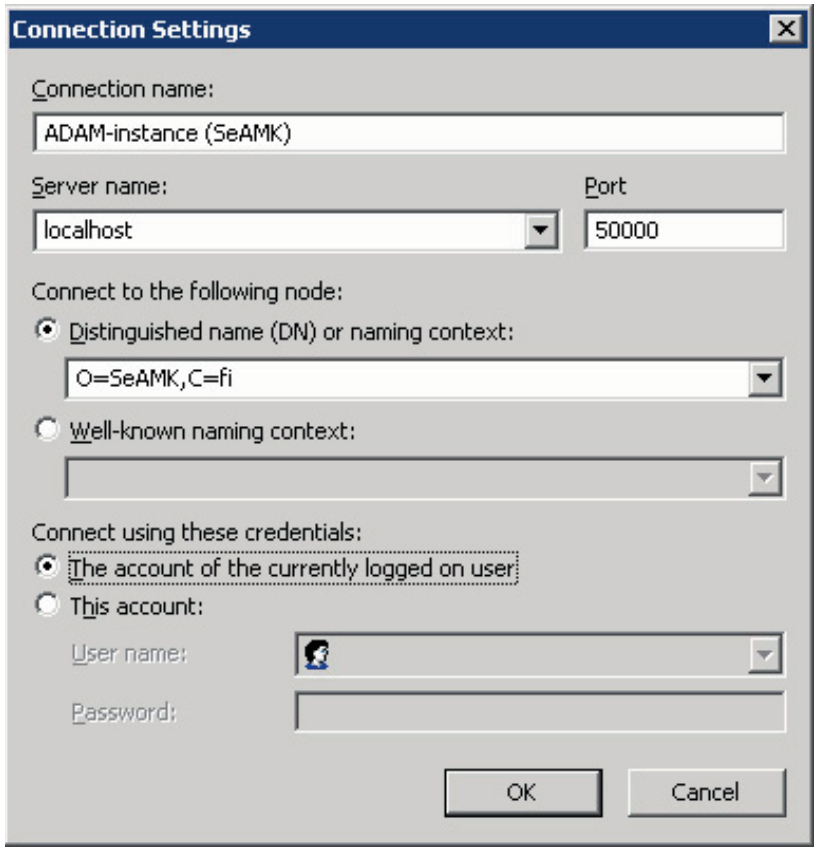
5.1 Active Directory Application Moden asennus

ADAM on vapaasti ladattava LDAP-hakemistotuote, jonka voi ladata Microsoftin sivuilta. ADAM:in asennustiedosto on nimeltään ADAMretailX86.exe. Asennuksen aikana joudutaan määrittämään muutamia arvoja ADAM-ilmentymälle. Asennusvelho pyytää arvoja seuraaviin ominaisuuksiin:

- Asennus ADAM:in ja hallintatyökalujen kanssa vai pelkät hallintatyökalut.
- Replikoidaanko toisesta ADAM-ilmentymästä vai tehdäänkö uusi ilmentymä.
- ADAM-ilmentymän nimi.
- Halutun LDAP- ja SSL-portin numerot.
- Halutun hakemisto-osion nimi.
- ADAM-ilmentymän datahakemisto ja datan palautushakemiston sijainnin.
- Palvelun päätilin.
- ADAM:in pääylläpitäjäntilin.
- Halutut skeematiedostot. Skeematiedostot ADAM:in asennuspaketissa ovat MS-InetOrgPerson.LDF, MS-User.LDF, MS-AZMan.LDF ja MSUserProxy.LDF.

Asennuksessa annetaan seuraavat arvot. Palvelimelle asennetaan ADAM ja hallintatyökalut. Uusi ADAM-ilmentymä luodaan. ADAM-ilmentymälle annetaan nimeksi LDAPDirectory. LDAP-portiksi annetaan portti numero 50000. SSL-portiksi annetaan portti numero 50001. Hakemisto-osion nimeksi annetaan O=SeAMK,C=fi. Datakansioksi ja datan palautuskansioksi ADAM:in asennus ehdottaa C:\Program Files\Microsoft ADAM\LDAPDirectory\data. Vakiokansioarvot ovat soveliaat, koska asennus on suositeltavaa C:\Program Files\Microsoft ADAM\ -kansioon. Palvelun päätiliksi kannattaa asettaa tili, joka löytyy aktiivihakemistostakin. ADAM:in Administrators -ryhmään kannattaa asettaa käyttäjä, joka löytyy myös aktiivihakemistosta. Tämä helpottaa ADAM:in hallintaa siinä määrin, että voidaan ottaa yhteys ADAM-ilmentymän pääresursseihin mistä vain. Administrators-ryhmään voidaan lisätä käyttäjiä myöhemminkin. Skeematiedostoista kannattaa valita kaikki, koska voi tulla tilanne, jolloin tarvitaan tiettyä skeema-attribuuttia. Näiden arvojen antamisen jälkeen ADAM:in asennus alkaa. Asennuksen jälkeen palvelimen uudelleenkäynnistys on suositeltavaa.

ADAM:in ilmentymän käynnistys voidaan varmistaa ADAM ADSI Edit -työkalulla. Työkalu löytyy asennuksen jälkeen Käynnistä-valikosta nimellä ADAM ADSI Edit. ADAM-ilmentymään yhdistettäessä määritetään muutama arvo. Kuvio 9 näyttää esimerkkiyhdistämisen arvot, kun yhteys otetaan palvelimelta käsin.



Kuvio 9. Esimerkkiyhdistämisarvot ADAM-ilmentymään ADAM ADSI Editillä.

Kuviossa 9 annetaan yhteyden nimeksi halutun palvelimen nimi. Yhteyttä otettaessa palvelimelta voidaan käyttää localhost-määrettä. Yhteys otetaan tällöin tietokoneeseen, johon on kirjauduttu. Muulloin tässä voidaan käyttää IP-osoitetta, DNS- tai NetBios-nimeä. Tämä jälkeen määritetään hakemiston portti. Seuraavaksi annetaan hakemisto-osion nimi. Tämä nimi on annettu ADAM:in asennuksen aikana. Hyvin tunnettuja nimeämisiä voidaan käyttää, kun yhdistetään hakemiston RootDSE-, konfiguraatio- (Configuration) tai skeemaosioon (Schema). Seuraavaksi määritellään, millä käyttäjänimellä hakemistoon otetaan yhteys. ADAM ADSI Edit -työkalua voidaan käyttää eri objektien lisäämiseen, muokkaamiseen ja selaamiseen LDAP-hakemistossa.

5.2 ADAM:in skeemaosion laajennus

ADAM:in skeemaosiota päätettiin laajentaa funetEduPerson-skeemalla. FunetEduPerson-skeema sisältää attribuutteja, jotka ovat nähty tarpeellisiksi korkeakoulumaailmassa. FunetEduPerson-skeematiedosto on saatavissa CSC:ltä (CSC, [viitattu: 1.11.2005]). FunetEduPerson-skeemaa muokattiin X.500-muotoon. FunetEduPerson-skeema on muunnetussa muodossa liitteestä 3. FunetEduPerson-skeemaa muokattaessa on käytetty taulukkoa, joka löytyy Inside Active Directory -kirjan kotisivuilta (Inside Active Directory, [viitattu 1.11.2005]). Tästä taulukosta löytyy arvot SYNTAX-attribuutin muuntamiseen attributeSyntax- ja oMSyntax-muotoon. Esimerkissä 1 on kuvattu kuinka opiskelijanumero-attribuutti muuntuu X.500-muotoon.

Esimerkki 1. Esimerkki skeema-attribuutin muunnosta X.500-muotoon.

```
Attribuutti alkuperäisessä muodossaan:
attributetypes: (1.3.6.1.4.1.16161.1.1.10
        NAME 'funetEduPersonStudentID'
        DESC 'student number or code'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

Kyseinen attribuutti muuntuu seuraavasti:
dn: CN=funetEduPersonStudentID,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonStudentID
attributeID: 1.3.6.1.4.1.16161.1.1.10
```

attributeSyntax: 2.5.5.12
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonStudentID
adminDescription: student number or code
description: student number or code
oMSyntax: 64
lDAPDisplayName: funetEduPersonStudentID
systemOnly: FALSE

Lopulliseen funetEduPerson-skeemaan päätettiin lisätä kaksi attribuuttia funetEduPersonAffiliation ja funetEduPersonGroup. FunetEduPersonAffiliation-attribuutti on henkilön rooli organisaatiossa, eli onko henkilö opiskelija vai henkilökuntaan kuuluva. FunetEduPersonGroup-attribuutti on opiskelijan luokkatunnus. Kaikista attribuuteista löytyvät kuvaukset skeematiedostosta, joka on liitteestä 3.

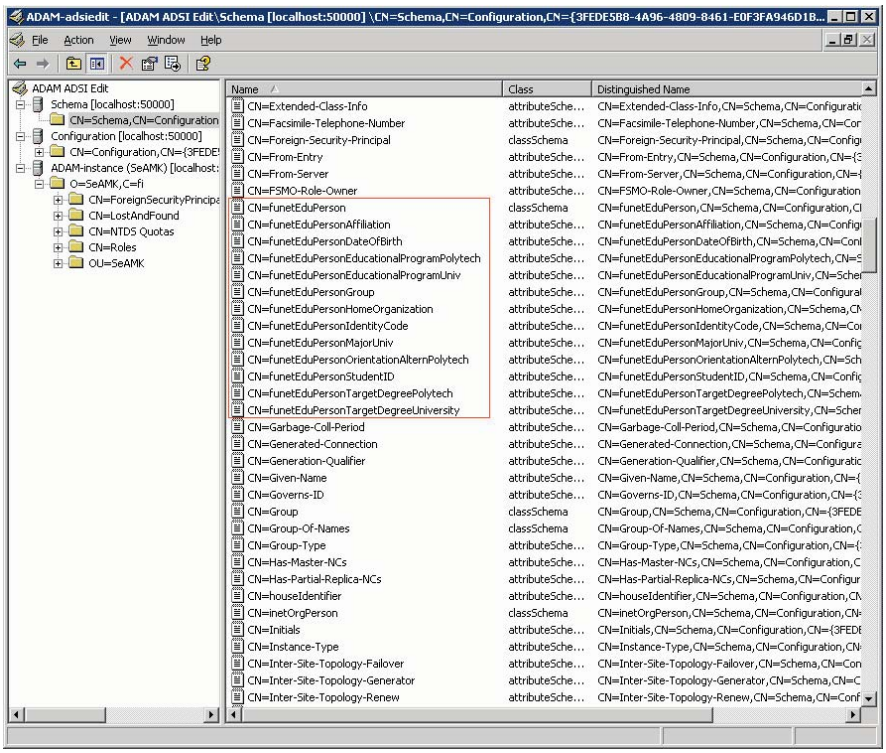
Skeeman vienti ADAM:iin onnistuu ldifde-työkalulla. Skeema-tietoja voidaan myös muokata ADAM ADSI Edit -työkalulla. Esimerkissä 2 on esitetty kuinka skeematiedosto viedään ADAM:iin ldifde-työkalulla.

Esimerkki 2. Skeematiedoston vienti ADAM:iin ldifde-työkalulla.

```
ldifde -i -f funetEduPerson_1_0.ldf -s localhost:50000 -k -j . -c
”CN=Schema,CN=Configuration,DC=X” #schemaNamingContext
```

Esimerkissä 2 olevat arvot tarkoittavat seuraavaa.
-i = Vientimoodi päälle (oletuksena on tuontimoodi päällä).
-f = Tiedoston nimi (tässä tapauksessa funetEduPerson_1_0.ldf).
-s = Palvelimen nimi ja portti (tässä tapauksessa localhost:50000).
-k = Vienti onnistuu välittämättä ”Object Already Exists” ja ”Constraint Violation” virheistä.
-j = Määritys lokitiedoston sijainnille (tässä tapauksessa ., eli kirjoitetaan siihen hakemistoon, mistä komento suoritetaan).
-c = Korvaa Distinguished Namen nimeämisessä olevat attribuutit hakemiston omilla attribuuteilla (tässä tapauksessa CN=Schema,CN=Configuratin, DC=X vaihtuu CN=Schema,CN=Configuration,CN={3FEDE5B8-4A96-4809-8461-E0F3FA946D1B} muotoon)

FunetEduPerson-skeeman lisääminen voidaan varmistaa ADAM ADSI Editillä ottamalla yhteys ADAM:in skeemaosioon. Lisätyt skeema-attribuutit näkyvät ADAM ADSI Editin selauskentässä. Kuviossa 10 on esitetty funetEduPerson-skeema attribuuttien laajentuminen ADAM:in skeemaosioon.



Kuvio 10. FunetEduPerson-skeeman laajentuminen ADAM:in skeemaosioon.

Kuviossa 10 on näkyvissä ADAM:in skeemaosio. Kuviossa on neliöity funetEduPerson-skeeman attribuutit. Kuviossa on näkyvillä myös muita skeemaosion attribuutteja. Kuvion vasemmassa reunassa näkyvät osiot, joihin on otettu yhteys. Näitä ovat kuviossa skeema-, konfiguraatio- ja ilmentymäosio (O=SeAMK,C=fi).

5.3 SQL Server 2000 asennus ja päivitys

MIIS 2003 vaatii toimiakseen SQL Server 2000 -tietokantapalvelimen. SQL Server 2000:ssa pitää olla asennettuna Service Pack 3 -päivitys. SQL Server 2000:sta on olemassa kolme pääversiota: Personal Edition, Standard Edition ja Enterprise Edition. MIIS 2003 vaatii toimiakseen joko Standard Editionin tai Enterprise Editionin.

SQL Server 2000 asennetaan MIIS 2003:n kanssa samalle palvelimelle, koska tiedonsiirto on nopeampaa MIIS 2003:n ja SQL Server 2000:n välillä ja tällöin verkkoviivettä ei ole. SQL Server 2000 asennus lähtee käyntiin asennus-CD:ltä valitsemalla autorun.exe, jos automaattinen käynnistys on poistettu käytöstä. Asennuksen aikana joudutaan määrittämään muutamia arvoja SQL Server 2000:n toimintaan liittyen.

SQL Server 2000:n asennusvelho pyytää arvot seuraaviin ominaisuuksiin:

- SQL Server 2000:n sijainti lokaaliin tietokoneeseen vai etätietokoneeseen.
- Luodaanko uusi ilmentymä vai muokataanko vanhaa. Jos SQL Server 2000 on jo asennettu, tällöin muokkaus on mahdollista tehdä, muuten ei.
- Organisaation tiedot.
- Osat, jotka SQL Server 2000:sta asennetaan. Mahdollista on valita asiakasohjelmistot, palvelin ja asiakasohjelmistot tai pelkkä yhteys.
- SQL Server 2000:n asennus voidaan tehdä tyypillisenä, minimaalisella tai mukautetulla kokoonpanolla.
- Palvelutilin määritykset.
- Autentikointitapa.
- Lisenssityyppi. Mahdollista valita joko prosessorilisenssi tai istuntomäärälisenssi. Molemmissa pitää määrittää arvo, montako prosessoria tai istuntoa on.

Asennuksessa annetaan seuraavat arvot. Asennus tehdään lokaalille tietokoneelle. SQL Server 2000:sta tehdään uusi ilmentymä. Organisaation tietoja kysyttäessä annetaan Seinäjoen koulutuskuntayhtymä organisaatioksi. SQL Server 2000:sta asennetaan palvelin- ja asiakasohjelmistot tietokoneelle. Asennuksen tyypiksi valitaan tyypillinen (Typical). Palvelupäätiliksi valitaan tili, joka löytyy aktiivihakemistosta ja samaa tiliä käytetään jokaiselle palvelulle. Nämä auttavat SQL Server 2000:n hallinnointia. Autentikointityypiksi valitaan Windows-autentikointi. Lisenssityypiksi valitaan prosessorilisenssi. Näiden arvojen antamisen jälkeen SQL Server 2000 asennus aloitetaan. Palvelimen uudelleenkäynnistäminen asennuksen jälkeen on suositeltavaa.

Service Pack 3a on SQL Server 2000:n kolmas suurempi päivitys, jota SQL Server 2000:een on julkaistu. Service Pack 4 on myös julkaistu SQL Server 2000 -palvelimelle, mutta kyseisellä päivityksellä on ollut yhteensopivuusongelmia MIIS 2003:n kanssa.

Service Pack 3a:n asennus on suoraviivainen. Asennuksen aikana kysytään autentikointitapa, jolla tietokantaan kirjaudutaan. Autentikointitavaksi valittiin Windows-autentikoinniksi SQL Server 2000:n asennuksessa.

5.4 Microsoft Identity Integration Server 2003 asennus

Uusimmassa MIIS 2003:n julkaisussa on integroitu Service Pack 1, joten sitä ei tarvitse erikseen asentaa. MIIS 2003:n asennus lähtee käynnistämällä Setup.exe MIIS 2003:n asennus-CD:ltä Setup-kansiosta, jos automaattinen käynnistys on poistettu käytöstä. MIIS 2003 vaatii asentuakseen muutamia eri arvoja. Asennuk-

sessa kysytään seuraavat arvot:

- Tehdäänkö täysi tai mukautettu asennus.
- SQL Server 2000 -palvelimen sijainti.
- Palvelun päätili.
- Ryhmät, jotka käyttävät MIIS 2003:a. Ryhmät luodaan palvelimen lokaaleihin ryhmiin.
- Talletuspaikka salausavaimille.

Asennuksessa annetaan seuraavat arvot. Asennus tehdään täydellisenä asennuksena (Complete). SQL Server 2000 asennettiin samalle palvelimelle, kun MIIS 2003 tullaan asentamaan, joten SQL Server 2000 löytyy tältä palvelimelta. Palvelun päätiliksi valitaan tili, joka löytyy aktiivihakemistosta. Tätä tiliä käytetään muun muassa salasanojen synkronointiin aktiivihakemiston ja MIIS 2003:n välillä. MIIS 2003:n asennusvelho ehdottaa seuraavia ryhmiä asennuksessa

- MIISAdmins (hallinnoijat)
- MIISOperators (operaattorit)
- MIISJoiners (yhdistäjät)
- MIISBrowse (selaajat)
- MIISPasswordSet (salasanojen hallinnoijat).

Ryhmät ovat näin kaikista loogisimmat. Vaihtoja ei tarvita. Kaikista tärkein ryhmä, jolla on täydet oikeudet MIIS 2003:n on MIISAdmins. Vähintään kaksi henkilöä on hyvä olla tässä ryhmässä. Näiden arvojen antamisen jälkeen asennus käynnistyy. Asennuksen loppuvaiheessa asennusvelho pyytää tallentamaan varmuuskopiot salausavaimista. Salausavaimet kannattaa tallentaa varmaan paikkaan. Salausavaimia tarvitaan mm. silloin, jos palvelunpäätiliä vaihdetaan. Salausavaimet voidaan myös hakea uudestaan Key Management Utility -ohjelmalla. Palvelimen uudelleenkäynnistys on suositeltavaa asennuksen jälkeen.

5.5 MIIS 2003 -skeeman laajennus

Skeeman laajennus MIIS 2003:ssa onnistuu Metaverse Designerillä, joka löytyy MIIS 2003:sta. Attribuuttien lisääminen onnistuu valitsemalla ensin objektityyppi ja sen jälkeen painamalla ”Add Attribute”-painiketta. Seuraavat attribuutit lisätään Person-objektityyppiin

- sAMAccountName (string)
- winhaAttendaceCode (string)
- winhaEducationalProgramPolytech (string)
- winhaEducPermission (string)
- winhaEnteringGroup (string)
- winhaInternetPermission (string)
- winhaIntranetPermission (string)

- winhaMarketingPermission (string)
- winhaOptionPreference (string)
- winhaOrganisationUnit (string)
- winhaPersonAffiliation (string)
- winhaStudentId (string)

Attribuuttien nimeäminen on suoritettu lähdehakemiston mukaan, koska silloin on helppo määrittää, mistä attribuutit tulevat. Kaikki lisätyt attribuutit ovat tietotyypiltään string eli merkkijonoja. Winhasta tulevien attribuuttien nimen edessä on tunnus winha. Muun muassa sAMAccountName-attribuutti tehdään Rules Extensioneiden avulla. SAMAccountName-attribuutissa yhdistetään yksikön tunnus ja winhaStudentId-attribuutti. SAMAccountName-attribuutista tulee mm. käyttäjän käyttäjätunnus. MIIS 2003:n skeemaan laajennettiin, koska kaikkia tarpeellisia attribuutteja ei löytynyt alkuperäisestä Person-objektityypistä.

5.6 Microsoft Visual Studio .Net 2003:n asennus

Rules Extensioneita ohjelmoitaessa tarvitaan ohjelmistokehitystyökalu. MIIS 2003 osaa integroitua Microsoft Visual Studio .Net 2003 -ohjelmistokehitystyökalun kanssa. Microsoft Visual Studio .Net 2003:n asennus on suositeltavaa palvelimelle, koska se helpottaa Rules Extensioneiden ohjelmointia, muokkausta ja testausta. Microsoft Visual Studio .Net 2003:sta ei kuitenkaan kannata asentaa kaikkia ohjelmointikieliä palvelimelle, koska MIIS 2003 ei tue muita ohjelmointikieliä kuin Visual Basic .Net 2003 ja Visual C# .Net 2003. Tässä opinnäytetyössä Rules Extensioneiden ohjelmointiin käytetään Visual C# .Net 2003:a.

Microsoft Visual Studio .Net 2003:n asennus lähteen käyntiin asennus-CD:ltä valitsemalla autorun.exe, jos automaattinen käynnistys on poistettu käytöstä. Asennuksen aikana määrityksiä tarvitsee tehdä suhteellisen vähän. Asennuksen aikana kysytään asennuspolku ja työkalut, joita halutaan asentaa Microsoft Visual Studio .Net 2003:n mukana. Ainoat ohjelmointikielet, jota kannattaa ottaa mukaan ovat Visual Basic .Net 2003 ja Visual C# .Net 2003, koska muita ohjelmointikieliä palvelimella ei tulla tarvitsemaan.

5.7 Management Agenttien luonti ja konfigurointi

Management Agenttien tehtävä MIIS 2003:n toiminnassa on hallinnoida lähde- ja kohdehakemistoja ja niiden Connector Spaceja. MIIS 2003:n tehdään Management Agentit Winhalle, aktiivihakemistolle ja ADAM:lle. Jokainen hakemisto tarvitsee vähintään yhden Mangement Agentin. Management Agentit ovat erilaisia jokaiselle hakemistolle. Tästä johtuen lähes jokainen konfiguraatio on erilainen.

5.7.1 Winhan Management Agent

Winha on opiskelijoiden primäärilähde. Tästä johtuen lähes kaikki attribuutit otetaan Winhasta. Winhasta tieto saadaan CSV-muodossa. CSV-tiedostossa attribuutit ovat eroteltuina puolipisteillä ja jokaisella objektilla on oma rivinsä. Sovellus, jolla tiedot saadaan CSV-muotoon Winhasta, on WPro2Ldap.exe. Attribuuteista, jotka Winhasta on mahdollista saada CSV-muotoon, on kuvattu liitteessä 4.

Management Agentin luonti lähtee käyntiin määrittelemällä uuden Management Agentin tyypin. Tiedon ollessa CSV-muodossa valitaan Delimited Text File, eli eroteltu tekstitiedosto. Muitakin mahdollisuuksia on valittavana tekstitiedostotyyppisille tiedostoille. Winhasta tulevaan CSV-tiedostoon pitää kuitenkin käyttää Delimited Text File -tyyppiä, koska muut ovat yhteensopimattomia sen kanssa. Erottelumerkiksi valitaan puolipiste (;). Seuraavaksi nimetään attribuutit. Attribuutit on nimetty samoilla nimillä, millä ne Winhasta tulevatkin. Tämä nimeäminen selkeyttää Management Agentteja. Winhan Management Agentin ankkuriattribuutiksi valitaan funetEduPersonStudentId. Ankkuriattribuutti on attribuutti, jolla tietty objekti yksilöidään metahakemistossa. Objektityyppi CSV-tiedostossa ja Metaversessä on Person. Attribuuttien nimet Winhan CSV-tiedostossa, Metaversessä ja niiden liikkumasuunta on kuvattu taulukossa 3.

Taulukko 3. Attribuuttien nimet ja liikkumasuunnat Winhan Management Agentilla.

CSV-tiedoston attribuutti	Suunta	Metaversen attribuutti
funetEduPersonStudentId	→	winhaStudentId
givenName	→	givenName
funetEduPersonEducationalProgram	→	winhaEducationalProgram
winhaOptionPreference	→	winhaOptionPreference
winhaOrganisationUnit	→	winhaOrganisationUnit
winhaEnteringGroup	→	winhaEnteringGroup
sn	→	sn
street	→	street
eduPersonAffiliation	→	winhaPersonAffiliation
telephoneNumber	→	telephoneNumber
mobile	→	mobile
winhaAttendanceCode	→	winhaAttendanceCode
winhaEducPermission	→	winhaEducPermission
winhaMarketingPermission	→	winhaMarketingPermission
postalAddress	→	postalAdress
postalCode	→	postalCode
preferredLanguage	→	preferredLanguage
winhaIntranetPermission	→	winhaIntranetPermission
funetEduPersonStudentId, winhaOrganisationUnit	→	sAMAccountName (Rules Extension)
givenName,sn	→	displayName (Rules Extension)
givenName	→	cn (Rules Extension)

Winhan CSV-tiedostosta pelkästään tuodaan tietoa Metaverseen, koska Winhan tietokantaan ei päästä suoraan käsiksi ja CSV-tiedostoon ei kannata päivittää mitään, koska päivitys tapahtuu erillisellä sovelluksella (WPro2Ldap.exe). Attribuuteista kolme viimeistä taulukossa 3 luodaan Rules Extensionin avulla. Toteutuksen aikana tehdyistä Rules Extensioneista on enemmän luvussa 5.8. Salasanan hallinnassa annetaan Management Agentille oikeus asettaa salasana ensimmäisellä kerralla (Set only).

5.7.2 Aktiivihakemiston Management Agent

Aktiivihakemistoon luodaan ja muokataan käyttäjäobjekteja Management Agentilla. Metahakemistosta viedään Winhasta saatujen attribuuttien perusteella käyttäjäobjekti tai muokataan vanhaa. Kaikkia attribuutteja, mitä Winhasta saadaan, ei viedä aktiivihakemistoon. Mangement Agentin tyyppi on Active Directory. Objektityyppi Metaversessä on Person ja aktiivihakemistossa User. Taulukko 4 esittää attribuuttien liikkumasuuntaa ja attribuutteja, jotka viedään aktiivihakemistoon.

Taulukko 4. Attribuuttien nimet ja liikkumasuunnat aktiivihakemiston Management Agentilla.

Aktiivihakemiston attribuutti	Suunta	Metaversen attribuutti
sAMAccountName	←	sAMAccountName
displayName	←	displayName
sn	←	sn
mobile	←	mobile
postalAddress	←	postalAddress
postalCode	←	postalCode
preferredLanguage	←	preferredLanguage
street	←	steet
telephoneNumber	←	telephoneNumber
givenName	←	cn
userAccountControl	←	winhaAttendanceCode (Rules Extension)
userPrincipalName	←	sAMAccountName (Rules Extension)

Metahakemistosta viedään tietoa aktiivihakemistoon. UserPrincipalName- ja userAccountControl-attribuuttit luodaan Management Agent Rules Extensionin avulla. Ensimmäisellä kerralla, kun käyttäjäobjekti luodaan aktiivihakemistoon, annetaan Metaversen Rules Extensionin avulla käyttäjälle salasana, pakotettu salasanan vaihto laitetaan päälle ja asetetaan tilin vanhentumisaika.

5.7.3 ADAM:in Management Agent

ADAM:iin luodaan ja muokataan käyttäjäobjekteja Management Agentilla. Management Agentin tyyppi on Active Directory Application Mode (ADAM). Objektin tyyppi ADAM:issa on funetEduPerson ja Metaversessä Person. Taulukko 5 esittää attribuuttien liikkumasuuntaa ja attribuutteja, jotka viedään ADAM:iin.

Taulukko 5. Attribuuttien nimet ja liikkumasuunnat ADAM:in Management Agentilla.

ADAM:in attribuutti	Suunta	Metaversen attribuutti
displayName	←	displayName
funetEduPersonStudentId	←	winhaStudentId
funetEduPersonGroup	←	winhaEnteringGroup
funetEduPersonHomeOrganization	←	winhaOrganisationUnit
givenName	←	givenName
mobile	←	mobile
postalAddress	←	postalAddress
postalCode	←	postalCode
sn	←	sn
street	←	street
telephoneNumber	←	telephoneNumber
funetEduPersonAffiliation	←	winhaPersonAffiliation
funetEduPersonEducationalProgram-Polytech	←	winhaEducationalProgram-Polytech

Metahakemistosta viedään tietoa ADAM:iin. Kaikki attribuutit viedään suoraan, joten Management Agentille ei tässä tapauksessa Rules Extensioneita tarvita. Ensimmäisellä kerralla, kun käyttäjäobjekti luodaan ADAM:iin, annetaan Metaversen Rules Extensionin avulla käyttäjäobjektille salasana ja pakotettu salasanan vaihto laitetaan päälle.

5.8 Rules Extensioneiden ohjelmointi

Rules Extensioneita ohjelmoitiin Management Agenteille kaksi ja Metaverselle kolme kappaletta. Lokitiedostojen luomiseen ohjelmoitiin oma DLL-kirjasto. Metaverse Rules Extensionit ohjelmoitiin aktiivihakemistolle ja ADAM:lle. Kolmas Metaverse Rules Extension ohjelmoitiin kahden edellisen Metaverse Rules Extensionin yhdistämiseen, koska MIIS 2003 ei voi käyttää, kun yhtä Metaverse Rules Extensionia. Käytettäessä monia eri hakemistoja on suositeltavaa tehdä Metaverse Rules Extensionit erikseen, koska se lisää loogisuutta, helpottaa virheiden hakua ja hallinnointia. Rules Extensioneiden ohjelmointiin käytettiin Visual C# .Net 2003 -ohjelmistokehitystyökalua. Rules Extensioneita tehdessä on käytetty apuna

Microsoftin MSDN-kirjastoa (MSDN-kirjasto, [viitattu 1.11.2005]). Tässä luvussa käydään läpi ensin Management Agenttien Rules Extionsionit ja sen jälkeen Metaversen Rules Extensionit. Management Agenttien koodissa käytetään csentry- ja mventry-luokkia. Csentry-luokalla viitataan hakemiston merkintöihin ja mventry-luokalla Metaversen merkintöihin.

5.8.1 Management Agent Rules Extensionit

Management Agent Rules Extensionit ohjelmoitiin aktiivihakemiston sekä Winhan Management Agenteille. Management Agent Rules Extensioneiden tehtävä on mm. luoda attribuutteja objekteille hakemistoihin ja Metaverseen. ADAM:in Management Agentille ei ohjelmoitu Rules Extensioneita, koska tällaista tarvetta ei ilmennyt.

Winhan Management Agentin Rules Extensionilla luodaan displayName-, sAMAccountName- ja cn-attribuutit tuodessa tietoa Metaverseen. Näihin operaatioihin käytetään MapAttributesForImport-metodia. DisplayName-attribuutti koostuu kahdesta eri Winhan attribuutista, jotka ovat givenName ja sn. SAMAccountName-attribuutti koostuu yksikön tunnuksesta ja henkilön opiskelijanumerosta. Seinäjoen ammattikorkeakoulun yksikkötunnukset ja Winhasta saatavat yksikkötunnukset ovat liitteessä 5. Esimerkissä 3 on listaus, kuinka objektille luodaan displayName-attribuutti Metaverseen.

Esimerkki 3. DisplayName-attribuutin luonti Metaverseen.

```
case ”cd.person:givenName,sn->mv.person:displayName”:

    // Poimitaan etunimi givenName-attribuutista.
    for (int i = 0; i <= csentry[”givenName”].Value.Length; i++)
    {
        try
        {
            if (csentry[”givenName”].Value.Substring(i, 1).Equals(@” ”))
            {
                firstName = csentry[”givenName”].Value.Substring(0, i);
                break;
            }

            if (i == csentry[”givenName”].Value.Length-1)
            {
                firstName = csentry[”givenName”].Value;
                break;
            }
        }

        catch (Exception)
        {
            continue;
        }
    }

    // Annetaan displayNamelle arvo sn-attribuutista ja poimitusta firstName-muuttujasta.
    mventry[”displayName”].Value = csentry[”sn”].Value + ”, ” + firstName;
break;
```

Esimerkissä 3 käytetään for-silmukkaa ja Substring-metodia givenName-attribuutin jakamiseen. Koska givenName-attribuutissa voi olla etunimi ja toinen nimi, näistä arvoista tarvitaan vain etunimi. Tämä siksi, koska displayName-attribuutti on yleensä muotoa sukunimi, etunimi. For-silmukkaa pyöritetään niin kauan, kunnes löytyy välilyönti-merkki. Tämän jälkeen firstName-muuttujalle asetetaan Substring-metodia käyttäen givenName-attribuutista nollannesta merkistä i:n arvoon merkit, jolloin saadaan käyttäjän etunimi selville. FirstName-muuttuja on tietotyypiltään string, eli merkkijono. Jos toista nimeä ei ole asetettu Winhan givenName-attribuuttiin, niin tällöinkin for-silmukka onnistuu, koska if-lauseella testataan tämä tapaus. Kun firstName-muuttujaan on poimittu käyttäjän etunimi, niin se lähetetään Metaverseen yhdistämällä sn-attribuutti ja firstName-muuttuja. Metaversen displayName-attribuutti saa arvon muodossa sukunimi, etunimi.

SAMAccountName-attribuutti luodaan käyttäen yksiköntunnusta ja opiskelijanumeroa, jotka saadaan luotua käyttäen kahta Winhan CSV-tiedostosta tulevaa attribuuttia. Nämä attribuutit ovat funetEduPersonStudentId ja winhaOrganisationUnit. SAMAccountName-attribuutista luodaan käyttäjälle mm. käyttäjätunnus aktiivihakemistoon. Esimerkissä 4 on listaus kuinka eri yksiköiden sAMAccountName-attribuutti luodaan yksiköntunnuksesta Metaverseen.

Esimerkki 4. SAMAccountName-attribuutin luonti Metaverseen.

```
case ”cd.person:funetEduPersonStudentId,winhaOrganisationUnit->mv.person:sAMAccountName”:

    // Selvitetään mistä yksiköstä henkilö on ja asetetaan yksikko-muutujalle arvo tämän mukaan.
    if (csentry[”winhaOrganisationUnit”].IsPresent)
    {
        // Tekniikan yksikkö/Seinäjoki
        if (csentry[”winhaOrganisationUnit”].Value.Equals(”KTEKNIIKKA”))
        {
            yksikko = ”c”;
        }

        // Liiketalouden yksikkö/Seinäjoki
        else if (csentry[”winhaOrganisationUnit”].Value.Equals(”KKAUPPA”))
        {
            yksikko = ”d”;
        }


        // Informaatio- ja kommunikaatioteknologian yksikkö/Seinäjoki
        else if (csentry[”winhaOrganisationUnit”].Value.Equals(”KICT”))
        {
            yksikko = ”l”;
        }

        // Yksikkötunnusta ei anneta, jos yksikkö on jokin muu.
        else
        {
            yksikko = ””;
        }
    }

    // Luodaan sAMAccountName-attribuutti yksikön kirjaintunnuksesta ja opiskelijanumerosta.
    mventry[”sAMAccountName”].Value = yksikko+csentry[”funetEduPersonStudentId”].Value;
    break;
```

Esimerkkissä 4 on kuvattu kolmen eri yksikön tunnuksen luonti. Muidenkin yksiköiden tunnukset tehdään samalla tavalla. Esimerkissä tarkistetaan, että Winhan CSV-tiedostosta tulevan winhaOrganisationUnit-attribuuttilla on arvo. Tämän jälkeen tarkistetaan winhaOrganisationUnit-attribuutin arvo ja asetetaan yksikko-muuttuja sen mukaan. Yksikko-muuttuja on muotoa string, eli merkkijono. Tämän jälkeen annetaan Metaversen sAMAccountName-attribuutille arvoksi yksikko-muuttujan arvo ja opiskelijanumero yhdistettynä.

Metaversen cn-attribuutti (Common Name) luodaan Winhan CSV-tiedostosta käyttäen givenName-attribuuttia. Cn-attribuuttia käytetään mm. aktiivihakemistoon luodessa uutta objektia. Esimerkissä 5 on listaus, kuinka cn-attribuutti luodaan Metaverseen.

Esimerkki 5. Cn-attribuutin luonti Metaverseen.

```
case ”cd.person:givenName->mv.person:cn”:

    // Poimitaan etunimi givenName-attribuutista.
    for (int i = 0; i <= csentry[”givenName”].Value.Length; i++)
    {
      try
      {
          if (csentry[”givenName”].Value.Substring(i, 1).Equals(@” ”))
          {
                firstName = csentry[”givenName”].Value.Substring(0, i);
                break;
          }

          if (i == csentry[”givenName”].Value.Length-1)
          {
                firstName = csentry[”givenName”].Value;
                break;
          }
      }

      catch (Exception)
      {
          continue;
      }
    }

    // Annetaan cn-attribuutille arvoksi poimittu firstName-muuttuja.
    mventry[”cn”].Value = firstName;
    break;
```

Esimerkissä 5 cn-attribuutti poimitaan Winhan CSV-tiedostosta tulevasta givenName-attribuutista. GivenName-attribuutti voi sisältää myös henkilön toisen nimen. GivenName-attribuutista tarvitaan kuitenkin vain etunimi. GivenName-attribuutista erotellaan etunimi käyttäen for-silmukkaa ja Substring-metodia. For-silmukkaa pyöritetään niin kauan, kunnes vastaan tulee välilyönti tai rivi on lopussa. Osoittimen osoittaessa merkkijonossa olevaan välilyöntiin firstName-muutujalle annetaan arvo

käyttäen Substring-metodia. Osoittimen osoittaessa viimeistä merkkiä firstName-muuttujalle annetaan arvoksi koko givenName-attribuutti, koska tällöin henkilöllä ei ollut givenName-attribuutissa toista nimeä. FirstName-muuttuja on muotoa string, eli merkkijono. Lopuksi annetaan Metaversen cn-attribuutille arvoksi firstName-muuttujan arvo.

Aktiivihakemistoon viedessä tietoa luodaan kaksi attribuuttia Management Agent Rules Extensionin avulla. Attribuutit, jotka Rules Extensioneilla luodaan, ovat userPrincipalName ja userAccountControl. Näihin operaatioihin käytetään Map-AttributesForExport-metodia. UserPrincipalName-attribuutti luodaan käyttäen Metaversestä saatavaa sAMAccountName-attribuuttia. Esimerkissä 6 on listaus, kuinka userPrincipalName-attribuutti luodaan aktiivihakemistoon.

Esimerkki 6. UserPrincipalName-attribuutin luonti aktiivihakemistoon.

```
case ”cd.user:userPrincipalName<-mv.person:sAMAccountName”:

    csentry[”userPrincipalName”].Value = mventry[”sAMAccountName”].Value + ”@EPEDU.local”;
break;
```

Esimerkissä 6 annetaan arvo hakemiston userPrincipalName-attribuutille käyttäen Metaversen sAMAccountName-attribuuttia sekä @-merkkiä ja toimialueen tunnistetta. Nämä attribuutit yhdistettäessä, saadaan userPrincipalName-attribuutti aktiivihakemistoon. UserPrincipalName-attribuuttia käytetään käyttäjäobjektien yksiselitteiseen tunnistamiseen aktiivihakemiston toimialueissa ja metsissä.

UserAccountControl-attribuutti aktiivihakemistossa määrää mm. onko käyttäjätili aktiivinen vai ei. UserAccountControl-attribuutille asetetaan arvo Winhan CSV-tiedostosta tulevalla winhaAttendanceCode-attribuutin avulla. WinhaAttendance-Code-attribuutti sisältää käyttäjän läsnäolokoodin. Esimerkissä 7 on listaus, kuinka userAccountControl-attribuutti luodaan aktiivihakemistoon.

Esimerkki 7. UserAccountControl-attribuutin luonti aktiivihakemistoon.

```
case ”cd.user:userAccountControl<-mv.person:winhaAttendanceCode”:

    // Henkilön erotessa tai valmituessa tili disabloidaan muuten enabloidaan.
    if ((mventry[”winhaAttendanceCode”].Value.Equals(”VP”)) ||
       (mventry[”winhaAttendanceCode”].Value.Equals(”ER”)))
    {
        csentry[”userAccountControl”].Value = ”514”; // Disabled.
    }

    else
    {
        csentry[”userAccountControl”].Value = ”512”; // Enabled.
    }

break;
```

Esimerkissä 7 annetaan arvo userAccountControl-attribuutille riippuen, opiskeleeko henkilö vielä oppilaitoksessa vai ei. Tarkistus tehdään käyttäen if-ehtoa, jossa katsotaan onko winhaAttendanceCode-attribuutin arvo joko VP (valmistunut) tai ER (eronnut). Jos henkilö on valmistunut tai eronnut, niin tili asetetaan sellaiseen tilaan, että sitä ei voida käyttää. Tällaiseen tilaan tilin viennissä userAccountControl-attribuutti saa arvon 514, jolloin tiliä ei voida käyttää. Muuten userAccountControl-attribuutti saa arvon 512, joka tarkoittaa, että tili on aktiivinen.

5.8.2 Metaverse Rules Extensionit

Metaverse Rules Extensioneiden päätarkoitus on tiedon provisiointi Metaversestä kohdehakemistojen Connector Spaceen, eli luoda uusi käyttäjäobjekti tai muokata vanhaa kohdehakemiston Connector Spacessa. Metaverse Rules Extensioneita ohjelmoitiin kaksi kappaletta ja lisäksi näiden kahden yhdistämiseen ohjelmoitiin oma Metaverse Rules Extension. Näiden lisäksi ohjelmoitiin DLL-kirjasto, jolla lokitiedot luodaan provisiointioperaatioista. Lokeilla voidaan seurata MIIS 2003:n toimintaa ja kirjoittaa halutuista virheistä omat virhetiedostot. Esimerkissä 8 on listaus DLL-kirjaston luokasta, jolla lokit luodaan.

Esimerkki 8. DLL-kirjaston luokka, jolla lokit luodaan.

```
public class CLogRuleExtension {

        protected string File_Log;
        protected byte Threshold_Log;
        protected bool LogConfigured = false;

        public CLogRuleExtension() {

        }

        protected void Log(string Entry, byte byLevel) {

            if(!(this.LogConfigured)) {

                // Log-tiedosto tallennetaan MIIS 2003:n MaData-kansioon.
                Microsoft.MetadirectoryServices.Logging.Logging.SetupLogFile(this.File_Log,this.Threshold_Log);
                this.LogConfigured = true;
            }

            Microsoft.MetadirectoryServices.Logging.Logging.Log(Entry,true,byLevel);
        }

        protected void Log(string Entry) {

            this.Log(Entry,0);
        }
}
```

Esimerkin 8 luokan alussa alustetaan tarvittavat muuttujat. Muuttujat ovat muotoa protected, eli muuttuja eivät näy julkisesti. CLogRuleExtension-luokassa on kaksi metodia, jolla lokia käsitellään. Ensimmäisessä asetetaan lokitiedosto käyttäen

MetadirectoryServices-nimiavaruuden SetupLogFile-metodia, jolla saadaan määritettyä lokitiedoston nimi ja lokitaso. Lokitasolle voidaan määrittää arvo 0 - 10, joka tarkoittaa lokin tärkeyttä. Toisessa metodissa kirjoitetaan lokiin haluttu arvo. CLogRuleExtension-luokkaa käytetään Metaverse Rules Extensioneiden lokitapahtumiin.

Aktiivihakemiston ja ADAM:in Metaverse Rules Extensionit ovat samantyylisiä. Molemmat jakavat tietoa, mutta eri Mangement Agenttien Connector Spaceen. OU-jakelu toimii molemmissa Metaverse Rules Extensioneissa samalla tavalla, koska OU-hierarkioista haluttiin samankaltaiset. OU-jakelu on tehty Seinäjoen koulutuskuntayhtymän nykyisen aktiivihakemiston mukaisella tavalla. Tämä sen takia, koska nykyiset OU-haarat ovat loogiset. ADAM:in Metaverse Rules Extensionin Provision-metodi on liitteessä 6. Metaverse Rules Extensioneissa käytetään Provision-metodia tiedon provisiointiin kohdehakemistojen Connector Spaceen. Metaverse Rules Extensioneista asetetaan tietyt attribuutit ensimmäisellä kerralla, kun objekti syötetään hakemiston Connector Spaceen. Esimerkissä 9 on listaus aktiivihakemiston Metaverse Rules Extensionin Provision-metodista.

Esimerkki 9. Aktiivihakemiston Metaverse Rules Extensionin Provision-metodi.

```
void IMVSynchronization.Provision (MVEntry mventry)
{

    string Agent_Contributing = null;
    string Agent_Target = null;
    string DN = null;

    // Määritellään MA:t
    Agent_Contributing = @"Winha_MA"; // Winhan Management Agent (lähde).
    Agent_Target = @"AD_MA"; //Active Directoryn Management Agent (kohde).

    try
    {

        this.Log(@"Provisioidaan... ");

        this.Log(string.Concat("Objektin Tyyppi: ",mventry.ObjectType));

        // Tarkistetaan, että objektin tyyppi on Person.
        switch(mventry.ObjectType)
        {
            case @"person":
                break;

            default:
                return;
        }

        this.Log(string.Concat("Jakelu Agentti: ",mventry[@"cn"].LastContributingMA.Name));

        // Tarkistetaan, että Winhan Management Agent on viimeksi provisioinut objekteja AD:hen.
        if(string.Compare(mventry[@"cn"].LastContributingMA.Name,Agent_Contributing,true)!=0)
        {
            return;
        }
```

```
// Määritetään Management Agent, jolle syötetään tietoa.
ConnectedMA Agent = mventry.ConnectedMAs[Agent_Target];
this.Log(string.Concat(@"Kohde Agentti: ",Agent.Name));
int Connectors = Agent.Connectors.Count;

string yksikko = null;
string opiskelijat = null;

// Tehdään OU objektille winhaOrganisationUnitista riippuen.
if (mventry["winhaOrganisationUnit"].Value.Equals("KMAASEUTU"))
{
    yksikko = ",OU=Maa-Metsa";
    opiskelijat = ",OU=Opiskelijat,OU=Ilmajoki";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KMETSÄ"))
{
    yksikko = ",OU=Maa-Metsa";
    opiskelijat = ",OU=Opiskelijat,OU=Opetus,OU=Tuomarniemi";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KTEKNIIKKA"))
{
    yksikko = ",OU=Tekniikka";
    opiskelijat = ",OU=Opiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKAUPPA"))
{
    yksikko = ",OU=Liiketalous";
    opiskelijat = ",OU=Opiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KYRITTÄJÄ"))
{
    yksikko = ",OU=Yrittajyys";
    opiskelijat = ",OU=eOpiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KSOSIAALI"))
{
    yksikko = ",OU=Sosiaali-Terveys";
    opiskelijat = ",OU=Opiskelijat,OU=Kampusalue";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KTERVEYS"))
{
    yksikko = ",OU=Sosiaali-Terveys";
    opiskelijat = ",OU=Opiskelijat,OU=Koskenala";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKULTTUURI"))
{
    yksikko = ",OU=Kulttuuri-muotoilu";
    opiskelijat = ",OU=AMK,OU=Opiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKULTKONS"))
{
    yksikko = ",OU=Kulttuuri-muotoilu";
    opiskelijat = ",OU=Opiskelijat,OU=konservaattorit";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKULTKUTU"))
{
    yksikko = ",OU=ICT";
    opiskelijat = ",OU=Kulttuuri,OU=Opiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KRAVITSEMI"))
{
    yksikko = ",OU=Ravitsemisala";
    opiskelijat = ",OU=AmkOpiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KICT"))
{
    yksikko = ",OU=ICT";
```

```
                // Katsotaan, onko henkilö TIKO vai TITE ryhmästä.
                for (int i = 0; i < mventry["winhaEnteringGroup"].Value.Length; i++)
                {
                    try
                    {
                        if   (mventry["winhaEnteringGroup"].Value.Substring(i, 4).Equals("TIKO"))
                        {
                            opiskelijat = ",OU=TIKO,OU=Opiskelijat";
                            break;
                        }

                        if   (mventry["winhaEnteringGroup"].Value.Substring(i, 4).Equals("TITE"))
                        {
                            opiskelijat = ",OU=TITE,OU=Opiskelijat";
                            break;
                        }

                        else
                        {
                            opiskelijat = ",OU=Muut,OU=Opiskelijat";
                        }
                    }

                    catch (Exception)
                    {
                        continue;
                    }
                }
            }


try
{

        if (mventry["winhaAttendanceCode"].Value.Equals("VP"))
        {
                // Henkilön valmistuessa, tili siirretään.
                DN = "CN="+mventry["sn"].Value+"\\, "+mventry["cn"].Value+","+"OU=Valmistuneet,OU=Disabloidut"+
                        opiskelijat+yksikko+",OU=SeAMK,DC=EPEDU,DC=local";
        }

        else if (mventry["winhaAttendanceCode"].Value.Equals("ER"))
        {
                // Henkilön erotessa, tili siirretään.
                DN = "CN="+mventry["sn"].Value+"\\, "+mventry["cn"].Value+","+"OU=Eronneet,OU=Disabloidut"+
                        opiskelijat+yksikko+",OU=SeAMK,DC=EPEDU,DC=local";
        }

        else
        {
                // Luodaan DN-string normaalille objektille.
                DN = "CN="+mventry["sn"].Value+"\\, "+mventry["cn"].Value+","+"OU="+mventry["winhaEnteringGroup"].Value+
                        opiskelijat+yksikko+",OU=SeAMK,DC=EPEDU,DC=local";
        }

}

// Kirjataan mahdolliset virheet lokiin.
catch(Exception cException)
{

        this.Log(string.Concat(@"Virhe: ",cException.Message));
        return;
}

this.Log(string.Concat(@"Provisioidaan ",mventry[@"cn"].Value));

// Määritetään Distinguished Name objektille DN-stringistä.
ReferenceValue DistinguishedName = Agent.CreateDN(DN);

this.Log(string.Concat(@"DN: ",DistinguishedName.ToString()));

CSEntry csentry = null;

switch(Connectors)
                {

                    case 0:
                        this.Log(@"Luodaan Uusi Connectori.");
                        string sObjectType = null;

                    // Tarkistetaan objektityyppi. Metaversestä tulevan objektin tyyppi pitää olla Person.
                    switch(mventry.ObjectType)
                    {
```

```
                    case @"person":
                        // Määritetään hakemiston objektityypiksi user.
                        sObjectType = "user";
                        break;

                    default:
                        throw new UnexpectedDataException("Odottamaton Objekti Tyyppi.");

                }
                    // Tallennetaan uusi Connectori AD:n Connector Spaceen.
                    csentry = Agent.Connectors.StartNewConnector(sObjectType);
                    csentry.DN = DistinguishedName; // Määritetään Distinguished Name objektille.
                    csentry["unicodePwd"].Values.Add("P@ssW0rd"); // Ensimmäinen salasana käyttäjälle.
                    csentry["pwdLastSet"].Value = "0"; // Pakotettu salasanan vaihto.
                    csentry["AccountExpires"].Values.Add(DateTime.Now.Date.AddMonths(54).ToFileTime());
                    csentry.CommitNewConnector(); // Luodaan uusi Connectori.
                    break;

                case 1:
                    // Jos objekti on jo olemassa, niin muokataan sitä.
                    this.Log(@"Muokataan Olemassa Olevaa Connectoria.");
                    csentry = Agent.Connectors.ByIndex[0];
                    csentry.DN = DistinguishedName;
                    break;

                default:
                    throw new UnexpectedDataException("Moninkertainen Connectori Management Agentilla!");
            }
        }

        catch (MissingParentObjectException)
        {

            // Kirjoitetaan lokiin puuttuvien OU-haarojen DN:t OUCreatoria varten.
            Microsoft.MetadirectoryServices.Logging.Logging.SetupLogFile("MissingOUs.txt", 10);
            Microsoft.MetadirectoryServices.Logging.Logging.Log(DN, false, 10);
            // Vaihdetaan normaali loki takaisin.
            Microsoft.MetadirectoryServices.Logging.Logging.SetupLogFile("Provisioning.log", 10);
        }


        catch(Exception cException)
        {
            // Virheet kirjoitetaan lokiin.
            this.Log(((cException.Message == null)||(cException.Message == string.Empty))
                        ?@"Tuntematon virhe.":string.Concat("Virhe: ",cException.Message));
            throw cException;
        }
}
```

Esimerkin 9 alussa alustetaan tarvittavat muuttujat ja asetetaan lähde ja kohde Management Agenttien nimet muuttujille. Tämän jälkeen tarkistetaan objektityyppi, joka tulee Metaversestä. Objektin tyyppi pitää olla Person, jotta provisiointi tapahtuisi. Tämän jälkeen tarkistetaan, että Winhan Management Agent on ollut viimeisin, joka on provisioinut tietoa aktiivihakemistoon. Tämä on pakollinen toimenpide konfliktien välttämiseksi. Connectors.Count-metodilla saadaan selville, onko kyseinen käyttäjäobjekti jo kohdehakemiston Connector Spacessa. Tämän jälkeen tarkistetaan henkilön yksikkö winhaOrganisationUnit-attribuutista. WinhaOrganisationUnit-attribuutin avulla tehdään OU objektille henkilön yksiköstä riippuen. Seuraavaksi tarkistetaan, onko käyttäjä valmistunut (VP) tai eronnut (ER). Tarkistus tehdään käyttäen winhaAttendanceCode-attribuuttia. Jos henkilö on valmistunut tai eronnut, niin käyttäjäobjekti siirretään omaan OU:nsa. Jos käyttäjää ei vielä ole aktiivihakemiston Connector Spacessa, niin objektille asetetaan unicodePwd-attribuutti (salasana), pwdLastSet-attribuutti (arvo 0 on pakotettu salasanan vaihto) ja AccountExpires-attribuutti (tilin vanhentumisaika tässä 4,5 vuotta). Tämän jälkeen

uusi käyttäjäobjekti syötetään kohdehakemiston Connector Spaceen. Käyttäjän ollessa jo aktiivihakemiston Connector Spacessa sitä muokataan. MIIS 2003:n antaessa virheilmoitukseksi MissingParentObjectException-virheen, niin tällöin kirjoitetaan objektin DN-arvo omaan tiedostoonsa. Tämä sen takia, jotta Organizational Unit -provisioija voi käyttää tätä hyödykseen tehdessään puuttuvia OU-haaroja. MissingParentObjectException-virhe tarkoittaa sitä, että objektilta puuttuu vanhempiobjekti, jolloin objektia ei voi syöttää Connector Spaceen. Aktiivihakemiston ja ADAM:in Rules Extensioneiden toiminnan aikana kirjoitetaan koko ajan lokiin mitä tapahtuu. Lokien kirjoitus kannattaa ohjelmoida Rules Extensioneihin, koska tällöin voidaan seurata paremmin MIIS 2003:n toimintaa.

Metaverse Rules Extension, jolla suoritetaan kaikki Metaverse Rules Extensionit peräkkäin, on saatavilla MSDN-kirjastosta. Tämän Metaverse Rules Extensionin tarkoitus on lukea tiedostonimen arvon perusteella kaikki Metaverse Rules Extensionit MIIS 2003:n Extensions-kansiosta ja suorittaa ne. Tämän Metaverse Rules Extensionin käyttö on silloin pakollista, jos käytetään monesta eri DLL-tiedostosta tulevia Metaverse Rules Extensioneita.

5.9 Organizational Unit -provisioijan ohjelmointi

MIIS 2003 ei pysty luomaan Organizational Unitteja (OU) suoraan. Tähän tarpeeseen ohjelmoitiin komentorivisovellus, joka luo OU:t MIIS 2003:n virhelokista. Tämä komentorivisovellus oli välttämätöntä luoda dynaamisen OU-provisioinnin takia. Dynaaminen OU-provisiointi tarkoittaa sitä, että OU:t tehdään siinä vaiheessa, kun objektia luodaan kohdehakemistoon. Komentorivisovelluksen nimi on OUCreator. Sovellus ohjelmoitiin käyttäen Visual C# .Net 2003 -sovelluskehitystyökalua. Sovellus hoitaa uusien OU-objektien tuonnin MIIS 2003:een. Uudet objektit pitää tuoda MIIS 2003:een, jotta MIIS 2003 voi tehdä provisioinnin. Uudelleentuontiin käytetään MIIS 2003:sta tehtäviä skriptejä. Organizational Unit -provisioija lähtee käyntiin komennolla OUCreator.exe ”virhetiedosto.txt”. Virhetiedostossa on yksi Distinguished Name riviä kohti.

OUCreatorin päämetodissa (Main) luodaan createOU-olio. CreateOU-oliolla on metodina createOUtoDirectory ja ExecuteAndWait. CreateOUtoDirectory-metodi hoitaa varsinaisen OU:n tekemisen kohdehakemistoon. ExecuteAndWait-metodilla saadaan suoritettua sovelluksen ulkopuolisia komentoja, joita on hyödynnetty uudelleentuonnissa. OUCreator odottaa parametreja ohjelmaa käynnistettäessä. Esimerkissä 10 on listattu OUCreatorin Main-metodi.

Esimerkki 10. OUCreatorin Main-metodi.

```
static void Main(string[] args)
{
        try
        {
            // Luetaan tiedoston nimi käyttäjän antamana argumenttina (esim. c:\MIIS\OUVirhe.txt).
            StreamReader OUtextFileReader = new StreamReader(@args[0]);

            // Tehdään createOU-oliot hakemistojen arvoilla.
            createOU AD = new createOU(@"LDAP://192.84.185.53:389/", @"DC=EPEDU,DC=local");
            createOU ADAM = new createOU(@"LDAP://192.84.185.53:50000/", @"O=SeAMK,C=fi");

            string Rivi = "";

            // Luetaan rivi kerrallaan tiedostosta ja tehdään OU:t molempiin hakemistoihin.
            while (Rivi != null)
            {
                Rivi = OUtextFileReader.ReadLine();

                if (Rivi == null)
                {
                    break;
                }

                if (Rivi != null)
                    AD.createOUtoDirectory(Rivi);
                    ADAM.createOUtoDirectory(Rivi);
                }

            OUtextFileReader.Close(); // Suljetaan lukija.
            File.Delete(@args[0]); // Poistetaan virhetiedosto.

            // Suoritetaan Management Agentit uudelleen
            AD.ExecuteAndWait("cscript.exe", "AD_Import.vbs");
            ADAM.ExecuteAndWait("cscript.exe", "ADAM_Import.vbs");
            AD.ExecuteAndWait("cscript.exe", "Winha_FullSynchronization.vbs");

        }

        catch (FileNotFoundException)
        {
            Console.WriteLine("Ei tehtäviä OU-haaroja tai tiedostoa ei löydy.");
            return;
        }

        catch (IndexOutOfRangeException)
        {
            Console.WriteLine("Anna tiedosto, joka sisältää Distiguished Namet.");
            return;
        }
}
```

Esimerkin 10 alussa luodaan StreamReader-olio nimeltä OUtextFileReader, jolla luetaan tiedostoa. StreamReader saa parametrina käyttäjän antaman tiedoston sijainnin (esimerkiksi C:\MIIS2003\OUVirhe.txt). Tämän jälkeen luodaan aktiivihakemistoa ja ADAM:ia varten createOU-oliot. CreateOU-oliolle annetaan parametreina hakemiston LDAP-polku, portti sekä juuren osoite. Tämän jälkeen siirrytään while-silmukkaan sisälle ja luetaan tiedostoa rivi kerrallaan. While-silmukan sisällä tarkistetaan, että rivi ei ole tyhjä. Rivin ollessa tyhjä, while-silmukka lopetetaan. Jos rivi ei ole tyhjä, niin luodaan tiedostosta tulevan Distinguished Namen mukainen OU. MIIS 2003 kirjoittaa yhden Distinguished Namen yhdelle riville tiedostoon. Kun tiedosto on luettu loppuun, niin OUtextFileReader suljetaan ja virhetiedosto poistetaan. Virhetiedosto poistetaan seuraavaa virhetiedostoa varten, joka tulee MIIS 2003:sta. Näiden operaatioiden jälkeen suoritetaan kolme skriptiä cscript-sovelluksen avulla. Cscript on sovellus, jolla voidaan suorittaa skriptejä komentoriviltä. Skripteistä ja suoritusjärjestyksestä enemmän luvussa 5.10. Jos tiedostoa ei löydy, niin ohjelma tulkitsee sen niin, että tehtäviä OU-haaroja ei ole.

CreateOU-olio on sovelluksen sydän, koska se hoitaa varsinaiset operaatiot. CreateOUtoDirectory-metodi hoitaa Organizational Unittien luomisen kohdehakemistoihin. Kohdehakemiston osoite ja juuri annetaan createOU-oliolle parametreina oliota luodessa. CreateOUtoDirectory-metodi tarvitsee parametrina Distinguished Namen, joka luetaan Main-metodissa tiedostosta. Esimerkissä 11 on listattu createOUtoDirectory-metodi.

Esimerkki 11. CreateOUtoDirectory-metodi.

```
public void createOUtoDirectory(string distinguishedName)
{
    string OU = distinguishedName;   // Muuttuja koko DN:lle.
    DirectoryEntry objDIRECTORY;  // Sitomisobjekti.
    DirectoryEntry objOU;             // DirectoryEntry Objecti OU:n tekemiseen.
    string strOU;                     // Tehtävän OU:n string.
    string strPath;                   // Sitomispolku.
    string ldapServer;                // LDAP-palvelimen osoite.
    string OC;                        // Loppuarvot.
    string OUString = null;           // Muuttuja OU:n lisäämiseen strPath muuttujaan.

    // Bindstringin kokoaminen.
    ldapServer = server;
    OC = org;
    strPath = ldapServer+OC;

    // Sitominen.
    try
    {
        objDIRECTORY = new DirectoryEntry(strPath);
        objDIRECTORY.RefreshCache();
        Console.WriteLine("TEHDÄÄN OU OBJEKTILLE: "+distinguishedName+" PALVELIMELLE: "+strPath);
    }

    catch (Exception exe)
    {
        // Annetaan virheilmoitus, jos sitominen on epäonnistunut.
        Console.WriteLine("Virhe:    Sitominen epäonnistunut.");
        Console.WriteLine("          {0}", exe.Message);
        return;
    }

    // OU-arvojen erottelu stringistä
    int aPaikka = 0;          // Aloituspaikka OU arvolle (OU=).
    int lPaikka = 0;          // Lopetuspaikka OU arvolle (, tai tyhjä).
    int kerta = 0;            // Silmukan lopetusarvo 3.
    int viimeisin = 0;        // lPaikka muuttujan viimeisin arvo.
    int syottoKerta = 0;      // Ensimmäisellä kertaa eri parametrit annettava.

    // Lähdetään lukemaan merkkijonoa oikealta vasemmalle.
    for (int i = OU.Length; i >= 0; --i)
    {
        try
        {
            // Jos Substringin i:n kohta ja + 3 merkkiä on yhtäkuin OU=, niin jatketaan seuraavaan operaatioon.
            if (OU.Substring(i, 3).Equals("OU="))
            {
                // Aloituspaikka saa arvon i.
                aPaikka = i;

                // Lähdetään lukemaan i:n paikasta vasemmalta oikealle merkkejä.
                for (int j = i+1; j < OU.Length; ++j)
                {
                // Hypätään Escape-merkin yli (lisätään j:tä kahdella).
                if (OU.Substring(j, 1).Equals(@"\"))
                    {
                        j = j+2;
                    }

                    // Pilkun tullessa vastaan lopetetaan ja annetaan arvo lopetuspaikalle.
                    if (OU.Substring(j, 1).Equals(","))
                    {
                        lPaikka = j - aPaikka;
                        viimeisin = lPaikka;
                        break;
                    }
```

```
                    // Osoittimen ollessa lopussa annetaan arvot ja lopetetaan.
                    if (j == OU.Length-1)
                    {
                        lPaikka = j - (aPaikka-1);
                        viimeisin = lPaikka;
                        break;
                    }

                    // Osoittimen ollessa alussa lopetetaan ja annetaan arvot.
                    if (j == 2)
                    {
                        aPaikka = 0;
                        lPaikka = viimeisin+1;
                        kerta = 3;
                        break;
                    }
                }

                strOU = OU.Substring(aPaikka, lPaikka); // Määritetään tehtävä OU.

                // OU:n tekeminen
                try
                {
                    // Jos on syötetty hakemistoon ennemmin, niin tehdään uudestaan sitominen ja annetaan
                    // uusi bindstring.
                    if (syottoKerta == 1)
                    {
                        strPath = ldapServer+OUString+",”+OC;
                        OUString = strOU+",”+OUString; // Lisätään OUStringiä uudella OU:n arvolla.
                        objDIRECTORY = new DirectoryEntry(strPath);
                        objDIRECTORY.RefreshCache();

                        objOU = objDIRECTORY.Children.Add(strOU,
                            ”OrganizationalUnit”);
                        objOU.CommitChanges();
                    }

                    // Ensimmäisellä syöttökerralla ei tarvitse tehdä uudestaa sitomista.
                    if (syottoKerta == 0)
                    {
                        syottoKerta = 1;
                        OUString = strOU; // Tehdään ensimmäinen OUString.
                        objOU = objDIRECTORY.Children.Add(strOU,
                            ”OrganizationalUnit”);
                        objOU.CommitChanges();
                    }

                }

                // Napataan kaikki virheet ja jatketaan.
                catch (Exception)
                {
                    continue;
                }

            }

            if (kerta == 3) // Jos osoitin rivin alussa, niin lopetetaan.
                return;
        }

        // Napataan kaikki virheet ja jatketaan.
        catch (Exception)
        {
            continue;
        }
    }
}
```

Esimerkki 11 metodin alussa luodaan tarvittavat muuttujat operaatioita varten. Tämän jälkeen kootaan varsinainen bindstring eli merkkijono, joka koostuu palvelimen LDAP-osoitteesta ja juuresta. Jos palvelinta ei löydy, niin siitä ilmoitetaan käyttäjälle. Tämän jälkeen alustetaan integer- eli numeeriset muuttujat erotteluoperaatioita varten. Distinguished Namen (DN) lukemiseen on käytetty Substring-metodia, jolla saadaan luettua merkkijonosta osia. Tämän jälkeen lähdetään lukemaan DN:ää

oikelta vasemmalle, koska viimeinen OU pitää syöttää palvelimelle ensimmäisenä. Tämä tarkoittaa sitä, että LDAP-polku koostuu hierarkisesti alhaalta ylöspäin. Ensin etsitään DN:stä arvo OU=. Kun arvo OU= on löydetty, niin lähdetään lukemaan DN:ää vasemmalta oikealle siitä kohdasta, josta OU=-arvo löytyi. Escape-merkki löydettäessä hypätään sen yli ja lisätään for-silmukan laskuarvoa kahdella. Escape-merkki DN:ssä tarkoittaa seuraavaa merkkiä, jota ei huomioida. Escape-merkki DN-merkkijonossa on kenoviiva (\). Esimerkki tällaisesta on pilkku, koska pilkulla erotellaan eri arvot (esimerkiksi CN- ja OU-arvot). Pilkun tullessa vastaan annetaan muuttujille aloituspaikka ja lopetuspaikka, jotta saataisiin luettua ensimmäinen tehtävä OU. Osoittimen ollessa DN:n lopussa tai alussa tehdään myös sama operaatio. Tämän jälkeen tehdään varsinainen OU-merkkijono, joka syötetään hakemistoon. Merkkijonon tekemiseen käytetään Substring-metodia ja for-silmukasta saatuja numeerisia arvoja. Ensimmäinen syöttökerran ollessa kyseessä ei uudelleensitomista palvelimeen tarvita. Tämä pätee kuitenkin vain ensimmäiseen kertaan. Ennen syöttämistä tarkistetaan, onko syötetty aiemmin tietoa kohdehakemistoon. Tämän jälkeen tehdään OU hakemistoon. Tätä silmukkaa jatketaan kunnes kaikki OU:t DN-merkkijonosta on tehty kohdehakemistoon. Jos kohdehakemistossa on jo kyseinen OU, niin sen ohi hypätään eli ei tehdä kyseistä OU:ta kohdehakemistoon, vaan jatketaan seuraavaan.

ExecuteAndWait-metodin tarkoitus on suorittaa ulkoinen sovellus. Tätä ominaisuutta tarvitaan uudelleentuonnissa tietoa metahakemistoon. ExecuteAndWait-metodi vaatii tiedoston nimen, joka suoritetaan ja tiedostolle annettavat argumentit. Esimerkissä 12 on listaus ExecuteAndWait-metodista.

Esimerkki 12. ExecuteAndWait-metodi.

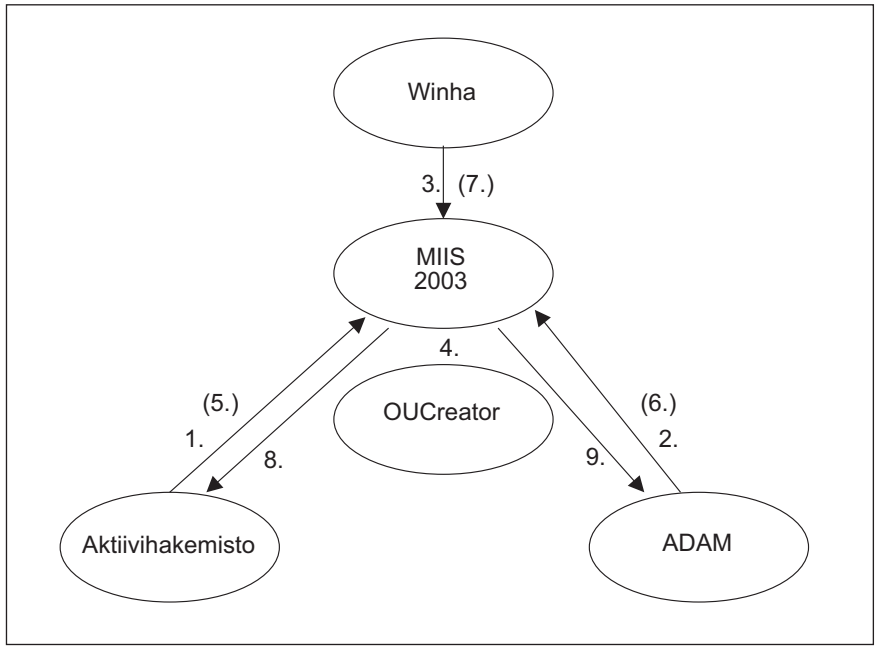
```
public void ExecuteAndWait(string fileToExecute, string Arguments)
{
    ProcessStartInfo command = new ProcessStartInfo(); // Muodostetaan uusi ProcessStarInfo()-olio.
    command.WindowStyle = ProcessWindowStyle.Normal; // Suoritetaan sovellus normaalilla ikkunalla.
    command.FileName = fileToExecute; // Annetaan FileName-atribuutille käyttäjän antama tiedoston nimi.
    command.Arguments = Arguments; // Annetaan Arguments-atribuutille käyttäjän antamat argumentit.

    Process prosessi = System.Diagnostics.Process.Start(command); // Suoritetaan prosessi...
    prosessi.WaitForExit();                                       // ...ja odotetaan sen päättymistä.
}
```

Esimerkin 12 metodissa luodaan uusi ProcessStartInfo-olio. Tämän jälkeen oliolle annetaan tyyli-arvo, jolla haluttu sovellus suoritetaan normaalissa ikkunakoossa. Tämän jälkeen oliolle annetaan tiedoston nimi ja argumentit, jotka suoritetaan. Tämän jälkeen ulkoinen sovellus käynnistetään ja odotetaan sen päättymistä.

5.10 Management Agenttien suoritusjärjestyksen suunnittelu ja luonti

Management Agenttien suoritusjärjestyksen pitää olla johdonmukainen, koska muuten tiedot eivät mene järjestelmällisesti oikein eri hakemistoihin. Suoritusjärjestyksen luomiseen on käytetty MIIS 2003:sta tehtäviä skriptejä sekä yhtä erillistä skriptiä. Skripti on mukaeltu käyttäen esimerkkiä, joka löytyy MSDN-kirjastosta. Skripteillä suoritetaan Management Agentteja komentoriviltä, jolloin saadaan tehtyä komentotiedosto, jolla voidaan suorittaa monta Management Agenttia peräkkäin. Järjestyksen suorittamiseen luotiin komentotiedosto (.cmd), josta varsinaiset komennot ovat jonossa. Komentotiedostoihin voidaan kirjata monta peräkkäistä komentoa. Suoritusaika on verrannollinen tietomäärään, jos tietoa on enemmän, niin prosessi kestää kauemman. Kuviossa 11 on kuvattu suoritusjärjestys.



Kuvio 11. Management Agenttien suoritusjärjestys.

Kuviossa 11 olevat numerot tarkoittavat Management Agenttien suoritusjärjestystä. Sulkeissa olevat numerot tarkoittavat sitä, että niitä ei välttämättä suoriteta. Lähdehakemistona toimii Winha ja kohdehakemistoina toimivat aktiivihakemisto ja ADAM. Suoritusjärjestys lähtee käyntiin aktiivihakemiston tietojen tuonnilla MIIS 2003:een (Full Import (Stage Only)). Seuraava vaihe on tuoda tiedot ADAM:ista MIIS 2003:een (Full Import (Stage Only)). Tämän jälkeen suoritetaan Winhan Management Agent, joka tuo tiedot MIIS 2003:een ja provisioi tiedot aktiivihakemistoon ja ADAM:iin (Full Import and Full Synchronization). Näiden jälkeen suoritetaan OUCreator-sovellus, joka havaitsee virhetiedostosta onko tarvetta luoda Organizational Unitteja.

Jos Organizational Unitteja tehdään, niin vaiheet 5 - 7 suoritetaan, muulloin ei. Vaiheissa 5 - 7 tiedot tuodaan uudelleen kohdehakemistoista ja suoritetaan Winhan Management Agent. Uudelleentuonti tehdään siksi, että saadaan uudet OU:t MIIS 2003:n Metaverseen. Tämän jälkeen tiedot viedään MIIS 2003:sta aktiivihakemistoon ja ADAM:iin (Export). Tämän prosessin aikana uudet objektit on tallennettu metahakemistoon ja kohdehakemistoihin. Jos muutoksia on tullut olemassa oleviin objekteihin, niin muokkausoperaatio tapahtuu samalla.

Komentotiedosto, joka on esimerkissä 13, on mukaeltu esimerkkistä, joka löytyy MSDN-kirjastosta. OUCreator-sovellus käyttää tietojen uudelleentuontiin MIIS 2003:sta tehtäviä skriptejä. Nämä skriptit saadaan tehtyä automaattisesti MIIS 2003:n suoritusprofiileista. Suoritusprofiilit ovat profiileja, joilla tehdään jotain kohde- tai lähdehakemistoon, esimerkiksi viedään tai tuodaan tietoa. Management Agentilla voi olla useampi suoritusprofiili. Management Agenttien suorituskomentotiedosto on listattu esimerkissä 13.

Esimerkki 13. Management Agenttien suorituskomentotiedosto.

```
set zworkdir=%~dp0
pushd %zworkdir%

rem Haetaan tiedot aktiivihakemistosta ja ADAM:sta (Import).
cscript RunMA.vbs /MA:”AD_MA” /RunProfile:”Import”
if {%errorlevel%} NEQ {0} (echo Error[%ErrorLevel%]: Komentotiedostossa ilmeni virhe) & (goto
exit_script)

cscript RunMA.vbs /MA:”ADAM_MA” /RunProfile:”Import”
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: Komentotiedostossa ilmeni virhe) & (goto
exit_script)

rem Haetaan tiedot Winhan CSV-tiedostosta ja provisioidaan (Full Import And Full Synchronization).
cscript RunMA.vbs /MA:”Winha_MA” /RunProfile:”Full Sync”
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: Komentotiedostossa ilmeni virhe) & (goto
exit_script)

rem Suoritetaan OUCreator-sovellus puuttuvien OU:iden luomiseen aktiivihakemistoon ja ADAM:
iin.
start /WAIT OUCreator.exe ”C:\Program Files\Microsoft Identity Integration Server\MaData\Miss-
ingOUs.txt”

rem Viedään tiedot kohdehakemistoihin (Export).
cscript RunMA.vbs /MA:”AD_MA” /RunProfile:”Export”
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: Komentotiedostossa ilmeni virhe) & (goto
exit_script)

cscript RunMA.vbs /MA:”ADAM_MA” /RunProfile:”Export”
```

```
if {%errorlevel%} NEQ {0} (echo Error[%errorlevel%]: Komentotiedostossa ilmeni virhe) & (goto
exit_script)

:exit_script
popd
endlocal
```

Esimerkissä 13 käytetään cscript-sovellusta, jolla voidaan suorittaa skriptejä komentoriviltä. RunMA.vbs on skripti, jolla voidaan suorittaa Management Agent tietyllä suoritusprofiililla. Komentotiedostossa testaan, että edellinen prosessi on onnistunut. Jos edellinen prosessi ei ole onnistunut, niin koko prosessi lopetetaan. Tämän sen takia, että virheet saadaan nopeasti selville, eikä konkreettisia tuhoja tule kohdehakemistoihin. Esimerkissä on määritetty työhakemisto ja se pakotetaan toimimaan koko suorituksen ajan kyseisessä hakemistossa. Eri skriptien lähdekoodit ovat liitteessä 7.

5.11 PCNS-palvelun asennus Domain Controllerille

Password Change Notification Service (PCNS) on palvelu, jolla salasanat voidaan replikoida melkein reaaliajassa aktiivihakemistosta MIIS 2003:n. Käyttäjän vaihtaessa salasanaa aktiivihakemistoon PCNS välittää salasanan MIIS 2003:lle, joka taas välittää sen halutuille hakemistoille. MIIS 2003:ssa pitää olla salasanojen synkronointiominaisuus päällä Management Agentilla sekä Metaversellä. PCNS-palvelu asennetaan Domain Controllerille. PCNS:n asennustiedosto löytyy MIIS 2003:n asennus-CD:ltä.

Asennuksen alussa joudutaan laajentamaan aktiivihakemistoon skeemaa. Skeeman laajennus onnistuu käyttäen PCNS:n asennustiedostoa. Asennustiedostolle pitää kuitenkin antaa argumentti, jotta skeeman laajennus tapahtuisi. Esimerkissä 14 on listattu aktiivihakemiston skeeman laajennuskomento.

Esimerkki 14. Aktiivihakemiston skeeman laajennuskomento.

```
PCNS.MSI SCHEMAONLY=TRUE
```

Esimerkin 14 komennossa on PCNS.MSI tarkoittaa asennustiedoston nimeä. MSI (Microsoft Installer) on Windowsin oma asennuspakettiformaatti. Loppu SCHEMAONLY=TRUE tarkoittaa sitä, että asennetaan pelkkä skeeman laajennus aktiivihakemistoon. Ilman tätä operaatiota ei PCNS:n asennus onnistu. Tämän jälkeen voidaan käynnistää asennus pelkästään suorittamalla PCNS.MSI. Asennusohjelma havaitsee, että skeeman laajennus on tehty ja asennusta voidaan jatkaa. PCNS-asennuksen aikana ei tarvitse tehdä mitään määrityksiä. Asennuksen jälkeen Domain Controller -palvelimen uudelleenkäynnistys on suositeltavaa.

PCNS-asennuksen jälkeen palvelu pitää konfiguroida, jotta salasanat replikoituisivat MIIS 2003 -palvelimelle. Konfiguroinnin aikana käytetään kahta työkalua. Nämä työkalut ovat Setspn ja Pcnscfg. Setspn on komentorivityökalu, jolla saadaan hallittua aktiivihakemiston SPN:iä (Service Principal Name). SPN:t ovat nimiä, joilla asiakasohjelmistot identifioi yksikäsitteisesti palveluinstanssit. Pcnscfg on komentorivityökalu, joka tulee PCNS-asennuksen mukana. Pcnscfg-komentorivityökalun tarkoitus on konfiguroida varsinainen PCNS-palvelu. Molempia tarvitaan, jotta salasanat replikoituisivat MIIS 2003:n. Esimerkissä 15 on listattu Setspn-komento.

Esimerkki 15. Setspn-komento.

```
Setspn -A PasswordReplication/MIIS2003.EPEDU.local EPEDU.local\MIISAdmin
```

Esimerkin 15 komennossa ”-A” tarkoittaa sitä, että uusi SPN-ilmentymä luodaan ja ”PasswordReplication/MIIS2003.EPEDU.local” tarkoittaa SPN:lle käyttäjän antamaa nimeä / täyttä tietokoneen toimialuenimeä. Loppu ”EPEDU.local\MIISAdmin” tarkoittaa tiliä, jolla SPN toimii. Tilillä, joka määritetään, pitää olla oikeus kirjautua palveluna (Log on as a Service) eli ohjelmat voivat käyttää tätä tiliä tällöin. Esimerkkissä 16 on listattu Pcnscfg-komento.

Esimerkki 16. Pcnscfg-komento.

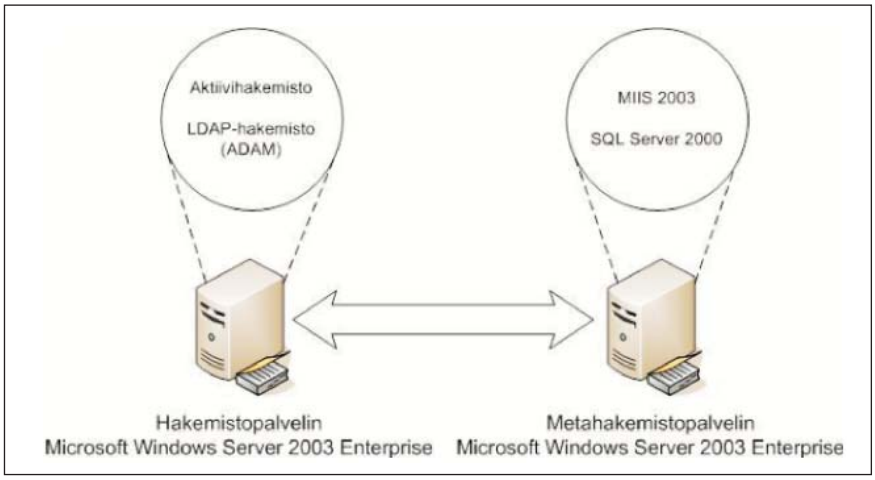
```
Pcnscfg ADDTARGET n:MIIS2003 a:MIIS2003.EPEDU.local /s:PasswordReplication/MIIS2003.
EPEDU.local /fi:”Domain Users” /f:3
```

Esimerkissä 16 olevat arvot tarkoittavat seuraavaa:
ADDTARGET = Lisätään uusi kohde.
n:MIIS2003 = Uuden kohteen nimi.
a:MIIS2003.EPEDU.local = Palvelimen osoite.
s:PasswordReplication/MIIS2003.EPEDU.local = SPN-nimi, joka määriteltiin aiemmin.
fi:”Domain Users” = Ryhmä, jonka salasanat lähetetään MIIS 2003:n.
f:3 = Formaatti, jolla ryhmät lähetetään. Tässä 3, eli NT 4.0 formaatti, joka on muotoa toimialue\käyttäjänimi.

Konfiguroinnin jälkeen, salasanan vaihdon tapahtuessa aktiivihakemistoon, salasanatiedot replikoituvat MIIS 2003 -palvelimelle. Salasanojen replikointia voi seurata tapahtumien valvonnasta (Event Viewer).

6 JÄRJESTELMÄN TESTAUS JA KÄYTTÖÖNOTTO

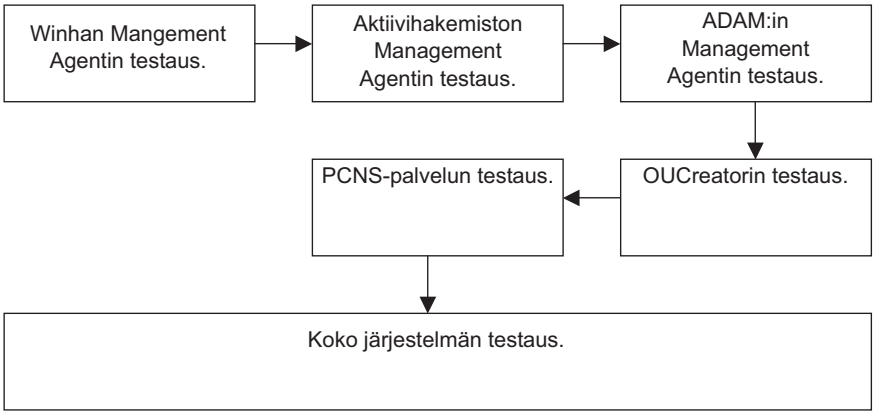
Testauksen tärkeys tämän tapaista järjestelmää otettaessa käyttöön on suuri. Testauksella saadaan suurimmat virheet pois varsinaisesta käyttöönotettavasta järjestelmästä. Testausta varten rakennettiin testiympäristö, jossa oli käytössä kaksi kappaletta Microsoft Windows Server 2003 Enterprise -käyttöjärjestelmillä varustettua palvelinta. Kuvio 12 kuvaa testiympäristön fyysistä rakennetta.



Kuvio 12. Testiympäristön fyysinen rakenne.

Aktiivihakemisto ja LDAP-hakemisto ovat fyysisesti samalla palvelimella, mutta ovat eri hakemistojärjestelmiä ja eivät ole riippuvaisia toisistaan. Hakemistopalvelimella toimi PCNS-palvelu, jolla salasanat replikoidaan metahakemistopalvelimelle. MIIS 2003 (metahakemisto) ja SQL Server 2000 ovat fyysisesti omalla palvelimella. Metahakemistopalvelimella on asennettu Visual Studio .Net 2003 -ohjelmistokehitystyökalu Rules Extensioneiden ohjelmointia ja testausta varten. Metahakemistopalvelimella on Winhan eri CSV-tiedostoja, joilla voidaan testata järjestelmän toimintaa. Hakemistopalvelin ja metahakemistopalvelin ovat yhteydessä toisiinsa synkronointiprosessin aikana.

Järjestelmän testauksessa käytettiin metodeina moduulitestausta, integrointitestausta ja järjestelmätestausta. Moduulitestauksessa on testattavana järjestelmän yksittäinen moduuli. Moduulitestauksella testattiin mm. Management Agenttien toimintaa. Integrointitestauksessa yhdistetään yhteen moduuleita tai moduuliryhmiä. Painopiste integrointitestauksessa on moduulien välisellä toimivuudella. Integrointitestauksella testattiin mm. Management Agenttien ja Rules Extensioneiden toimintaa. Järjestelmätestauksella testattiin koko järjestelmän toimintaa pienillä ja suurilla tietomäärillä. Kuviossa 13 on kuvattu, kuinka testausprosessi eteni. (Haikala, I. & Märijärvi, J. 2002, 287)



Kuvio 13. Testausprosessin eteneminen.

Testaus alkoi Winhan Management Agentin testauksella. Winhan Management Agentin pääpainopiste testauksessa oli tiedon tuonnissa ja attribuuttien luomisessa tiedon tuonnin aikana. Aktiivihakemiston testauksessa testattiin aktiivihakemiston Management Agentin tiedon vientiä varsinaiseen hakemistoon ja attribuuttien luontia viennin aikana, sekä aktiivihakemiston Management Agentin kykyä tuoda uutta tietoa Connector Spaceen. ADAM:in Management Agentin testauksessa testattiin ADAM:in Management Agentin tiedon vientiä varsinaiseen LDAP-hakemistoon, sekä Management Agentin kykyä tuoda uutta tietoa Connector Spaceen. Aktiivihakemiston ja ADAM:in Management Agenttien testauksessa testattiin myös tietojen muokkaantumista kohdehakemistoon Metaversestä. Metaverse Rules Extensioneiden tarkoitus on tietojen provisiointi Metaversestä Connector Spaceen. Metaverse Rules Extensioneiden testaus järjestyi samalla, kun eri Management Agentteja testattiin. Näiden operaatioiden jälkeen testatiin OUCreator-sovellusta, jonka tarkoitus on luoda OU:t kohdehakemistoihin MIIS 2003:n virhelokista. Tämän jälkeen testattiin PCNS-palvelu, jolla salasanat replikoidaan aktiivihakemistosta MIIS 2003:een melkein reaaliajassa. Testauksessa testattiin salasanojen replikoitumista aktiivihakemistosta MIIS 2003:een. Järjestelmän koko testauksessa testattiin järjestelmän toimivuus kuormittaessa järjestelmää pienillä ja suurilla tietomäärillä. Moduulitestausta suoritettiin koko testijärjestelmän rakennuksen ajan. Kaikkia moduulitestauksen tapahtumia on mahdotonta raportoida, koska niitä on niin paljon.

Testijärjestelmä suoriutui testeistä pääpiirteittäin hyvin. Muutamia virheitä kuitenkin havaittiin, mutta ne korjattiin. Virheet, jotka ilmenivät testausvaiheessa, on esitelty seuraavaksi korjauksineen.

Winhan Management Agentin ankkuriattribuutti. Alun perin testiympäristössä Winhalla oli kaksi ankkuriattribuuttia. Tämä muodostui virheeksi tilanteessa, jossa testattiin käyttäjän yksikön vaihtoa. MIIS 2003 loi toisen käyttäjän, joka oli toisessa yksikössä Metaversessä, eli sama käyttäjä oli Metaversessä kahteen kertaan. Asia ratkaistiin niin, että opiskelijanumero jätettiin ainoaksi ankkuriattribuutiksi.

Aktiivihakemiston ja ADAM:in Metaverse Rules Extensionin OU:n muodostamisessa yhden yksikön puuttuminen. Aktiivihakemiston ja ADAM:in Metaverse Rules Extensioneista puuttui kulttuurialan ja muotoilun yksikön attribuutti (KKULTKUTU). Attribuutti lisättiin if-ehtolistaukseen.

PCNS-palvelun replikointitili. Palvelun tili oli asetettu tietokoneelle vaikka tili pitää asettaa käyttäjälle. Tili asetettiin käyttäjälle.

Järjestelmän koko testaus ajastetulla ajosyklillä. Komentotiedossa esiintyi alun perin ongelma, jossa kaikki Management Agentit suoritettiin yhtä aikaa. Ongelma korjattin käyttäen RunMA.vbs-skriptiä, joka odottaa mm. Management Agentin suorituksen loppua.

Virheiden määrä jäi vähäiseksi testausvaiheessa, koska pienimmät virheet korjattin moduulitestauksen aikana. Järjestelmän asennus suunniteltiin tarkasti. Tämän toimenpiteen ansiosta virheet vähenivät todella paljon. Testauksessa saatiin kuitenkin selville suurimmat virheet. Testaus suunnitellaan yleensä järjestelmällisesti ja toteutetaan tämän sunnitelman mukaan. Näin toimittiin myös tässä tilanteessa. Tämä ei kuitenkaan paljasta kaikkia mahdollisia virheitä järjestelmästä. Mikään järjestelmä tai ohjelmisto ei ole virhevapaa. Mahdollisten virheiden tullessa vastaan ne kuitenkin korjataan.

Testiympäristöstä on helppo siirtyä varsinaiseen toimintaympäristöön, koska testiympäristö rakennettiin mukaellen Seinäjoen koulutuskuntayhtymän ympäristöä. Varsinaisessa toimintaympäristössä on oma palvelin LDAP-hakemistoa (ADAM) ja metahakemistoa (MIIS 2003) varten. SQL Server 2000 asennetaan metahakemistopalvelimelle. Metahakemistoon tullaan vielä yhdistämään WM-datan henkilöstöhallintojärjestelmä ja Datamarin opiskelijahallintojärjestelmä Studenta. Näiden kahden järjestelmän yhdistäminen tulee olemaan suhteellisen helppoa, koska tekniikka on selvillä ja tiedot saadaan CSV-muotoon myös näistä kahdesta järjestelmästä. Metahakemiston ja LDAP-hakemiston käyttöönotto sekä olemassa olevien hakemistojen yhdistäminen (aktiivihakemisto ja Winha) metahakemistoon Seinäjoen koulutuskuntayhtymässä tullaan tekemään tämän opinnäytetyön pohjalta.

7 JOHTOPÄÄTÖKSIÄ

Opinnäytetyön tavoitteena oli asentaa, konfiguroida ja ohjelmoida Seinäjoen koulutuskuntayhtymälle metahakemisto ja LDAP-hakemisto. Tavoitteena oli myös yhdistää kolme hakemistoa metahakemistoon. Nämä hakemistot olivat aktiivihakemisto, LDAP-hakemisto (ADAM) ja opiskelijahallintojärjestelmä Winha. Testausta varten rakennettiin testiympäristö, jolla voitiin testata kyseisen järjestelmän toimintaa samanlaisessa toimintaympäristössä kuin Seinäjoen koulutuskuntayhtymälläkin on. Testiympäristö on suositeltavaa rakentaa ennen varsinaista käyttöönottoa, koska tällöin järjestelmä on helppo käyttöönottaa varsinaiseen ympäristöön ja järjestelmä saadaan pois niin sanotuista ”lapsenkengistä”. Tämän tapaisen järjestelmän hyöty suurissa organisaatioissa on suhteellisen suuri, koska se helpottaa käyttäjähallintaa paljon. Opinnäytetyössä käsiteltiin metahakemistoa ja kolmea muuta hakemistoa, mutta kokonaiseen järjestelmään tullaan yhdistämään vielä muutama eri hakemisto, kuten opiskelijahallintojärjestelmä Studenta.

Opinnäytetyön alussa asetettiin tavoitteeksi aktiivihakemiston, LDAP-hakemiston (ADAM) ja Winhan yhdistäminen metahakemistoon. Nämä tavoitteet toteutuivat. Opinnäytetyö on antanut paljon tietoa metahakemiston toiminnasta, konfiguroinnista ja ohjelminnista. Erilaiset elektroniset hakemistot tulivat paremmin tutuiksi. Opinnäytetyöni aikana perehdyttiin paremmin Visual C# .Net -ohjelmointiin, joten uutta oppia kyseisestä ohjelmointikielestä saatiin. Käyttäjähallinta tuli tutummaksi hyödyntämällä erilaisia tekniikoita.

Vastaavanlaista opinnäytetyötä tai tutkimusta metahakemiston ja LDAP-hakemiston asentamisesta, konfiguroinnista ja ohjelmoinnista ei ole aiemmin tehty, joten lähdekirjallisuus tähän aiheeseen oli vähäinen. Eri hakemistoihin kirjallisuutta löytyi, mutta varsinaiseen metahakemistoon sitä ei löytynyt, joka oli kuitenkin suuri osa tätä opinnäytetyötä. Yksi vastaavanlainen järjestelmä on rakennettu Suomessa. Tämän järjestelmän on rakentanut Turun ammattikorkeakoulu. Turun ammattikorkeakoululla on kuitenkin erilaisempi ympäristö kuin Seinäjoen koulutuskuntayhtymällä.

Opinnäytetyön hyöty muille vastaavanlaisen ympäristön omaaville organisaatioille on suhteellisen suuri silloin, jos kyseisellä organisaatiolla on suunnitelmissa samantapaisen käyttäjähallintajärjestelmän rakentaminen. Hakemistot, joita opinnäytetyössä käsiteltiin, ovat laajasti käytettyjä. Esimerkiksi Microsoftin aktiivihakemisto on käytössä monessa suomalaisessa organisaatioissa.

Järjestelmää tullaan jatkokehittämään, koska Seinäjoen koulutuskuntayhtymällä on vielä hakemistoja joita tullaan yhdistämään metahakemistoon. Järjestelmän jatkokehityksessä on hyvä ottaa huomioon eri hakemistojen erilaisuudet ja mahdolliset ristiriidat primäärilähdehakemistojen välillä. Tästä hyvä esimerkki on henkilön kuuluminen henkilökuntaan ja opiskelijoihin. Tällöin tulee ottaa huomioon, kum-

pi on käyttäjän primäärilähde, koska saman henkilön tiedot tulevat kahdesta eri primäärilähteestä.

Opinnäytetyöprosessi kaikkiaan oli todella mielenkiintoinen. Aihe oli haastava ja antoi paljon uutta tietoa liittyen metahakemistoihin ja eri elektronisiin hakemistoihin. Ohjelmoititaidot kehittyvät Rules Extensioneita ohjelmoitaessa sekä DLL-kirjastojen ohjelmoinnin tuntemus parani. Käsitys käyttäjähallinnasta suurissa organisaatioissa syventyi. Käyttäjähallinta saattaa olla hyvin työlästä ja monimutkaista suuremmissa organisaatioissa suuren käyttäjämäärän vuoksi.

LÄHTEET

Active Directory Application Mode Technical Reference (Draft). 2004. Active Directory Application Moden tekninen referenssi. [Dokumentti]. [Viitattu: 20.8.2005]. Saatavissa: http://www.microsoft.com/downloads/details.aspx?familyid=96c660f7-d932-4f59-852c-2844b343f3e0&displaylang=en

Allen, R. & Lowe-Norris, A. 2003. Active Directory, 2nd Edition. Sebastopol: O'Reilly & Associates, Inc.

CSC. Tieteen tietotekniikan keskus CSC. [WWW-dokumentti]. [Viitattu 1.11.2005]. Saatavissa: http://www.csc.fi

Haikala, I. & Märijärvi, J. 2002. Ohjelmistotuotanto. 8. Painos. Helsinki: Talentum Media Oy.

IETF. The Internet Engineering Task Force. [WWW-dokumentti]. [Viitattu 20.10.2005]. Saatavissa: http://www.ietf.org/

Inside Active Directory. The Inside Active Directory book. [WWW-dokumentti]. [Viitattu 1.11.2005]. Saatavissa: http://www.kouti.com/

Introduction to Windows Server 2003 Active Directory Application Mode. 2005. Johdatus Windows Server 2003 Active Directory Application Modeen. [Dokumentti]. [Viitattu 2.9.2005]. Saatavissa: http://www.microsoft.com/windowsserver2003/techinfo/overview/adam.mspx

JMU - Directory Services. 2005. JMU - Hakemistopalvelut. [WWW-dokumentti]. [Viitattu 20.8.2005]. Saatavissa: http://www.jmu.edu/computing/systems/ds.shtml

Järvinen, P. & Järvinen, A. 2002. Tutkimustyön metodeista. Tampereen yliopisto: Opinpaja.

Key MIIS Concepts. MIIS:in avainkäsitteistö. [WWW-dokumentti]. [Viitattu 1.8.2005]. Saatavissa: http://www.activeidm.com/8627.1.43520/Key_MIIS_Concepts.html

Kouti, S. & Seitsonen, M. 2004. Inside Active Directory Second Edition. Boston: Addison-Wesley.

Metaconnections. Metayhteydet. Metaconnections kotisivu. [WWW-dokumentti]. [Viitattu 15.7.2005]. Saatavissa: http://www.metaconnections.co.uk/

Microsoft Identity Integration Server 2003 Developer Reference. 2005. Microsoft Identity Integration Server 2003:n kehittäjän referenssi. [WWW-dokumentti]. [Viitattu 2.8.2005]. Saatavissa: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/mmsdev/mms/about_rules_extensions.asp

Microsoft Identity Integration Server 2003 Help. Microsoft Identity Integration Server 2003 Help -dokumentti. [Dokumentti]. [Viitattu 3.8.2005].

Microsoft Identity Integration Server 2003 Home. Microsoft Identity Integration Server 2003:n kotisivu. [WWW-dokumentti]. [Viitattu 10.6.2005]. Saatavissa: http://www.microsoft.com/windowsserversystem/miis2003/default.mspx

Microsoft Identity Integration Server 2003 Product Overview. 2004. Microsoft Identity Integration Server 2003:n yleisesittely. [WWW-dokumentti]. [Viitattu 19.7.2005]. Saatavissa: http://www.microsoft.com/windowsserversystem/miis2003/evaluation/overview/default.mspx

MSDN-kirjasto. MSDN Home. [WWW-dokumentti]. [Viitattu 1.11.2005]. Saatavissa: http://msdn.microsoft.com/

Sedu. Seinäjoen koulutuskeskus. [WWW-dokumentti].[Viitattu 26.10.2005]. Saatavissa: http://www.sedu.fi/

SeAMK - yksiköt. Seinäjoen ammattikorkeakoulu - SeAMK - Yksiköt. [WWW-dokumentti].[Viitattu 26.10.2005]. Saatavissa: http://www.seamk.fi/yksikot/

Smit, R. Active Directory Applicatin Mode. [WWW-dokumentti]. [Viitattu 20.8.2005]. Saatavissa: http://dotnetjunkies.com/WebLog/ricksmit/archive/2004/02/19/7553.aspx

Technical Overview of MIIS 2003 with Service Pack 1. 2005. Tekninen yleisesittely MIIS 2003:sta, joka sisältää Service Pack 1:en. [Dokumentti]. [Viitattu 27.7.2005]. Saatavissa: http://www.microsoft.com/windowsserversystem/miis2003/techinfo/planning/miis.mspx

Webopedia. Online Computer Dictionary for Computer and Internet Terms and Definitions. [WWW-dokumentti]. [Viitattu 7.10.2005]. Saatavissa: http://www.webopedia.com/

Winhapro 4.5 LDAP. WM-Data. [Dokumentti]. [Viitattu 11.10.2005].

Zoomit Via and the Dedicated Metadirectory. 1998. Zoomit Via ja vakaumuksellinen metahakemisto. Windowsitpro kotisivu. [Verkkolehtiartikkeli]. [Viitattu 19.7.2005]. Saatavissa: http://www.windowsitpro.com/Article/ArticleID/3021/3021.html?Ad=1

Henkilökohtaiset tiedonannot

Koponen, E., tietojenkäsittelyn lehtori. Seinäjoen ammattikorkeakoulu. & Myllyaho, A., atk-pääsuunnittelija. Seinäjoen koulutuskuntayhtymä. Henkilökohtainen tiedonanto 23.9.2005.

Koponen, E., tietojenkäsittelyn lehtori. Seinäjoen ammattikorkeakoulu. Henkilökohtainen tiedonanto 23.5.2005.

Koponen, E., tietojenkäsittelyn lehtori. Seinäjoen ammattikorkeakoulu. Henkilökohtainen tiedonanto 16.9.2005.

Koponen, E., tietojenkäsittelyn lehtori. Seinäjoen ammattikorkeakoulu. Henkilökohtainen tiedonanto 1.11.2005.

Käenmäki, J., atk-pääsuunnittelija. Seinäjoen koulutuskuntayhtymä. Haastattelu 11.10.2005.

Myllyaho, A., atk-pääsuunnittelija. Seinäjoen koulutuskuntayhtymä. Sähköpostikeskustelu 7.10.2005.

Mäkelä, V-M., atk-tukihenkilö. Seinäjoen ammattikorkeakoulu. Haastattelu 7.9.2005.

Liite 1. Kokouksien päätökset ja pääkeskusteluaiheet.

1.4.2005

Kokouksessa kävimme lävitse, kuinka tiedot voitaisiin ajaa Winhasta LDAP-hakemistoon ja aktiivihakemistoon. Kokouksessa päätimme myös henkilöt, jotka valmistuvat tai eroavat siirtyvät omaan Organizational Unittiin. Ensimmäiset ehdotukset LDAP-hakemistosta, joka voisi olla Active Directory Application Mode. Mietimme myös synkronointijärjestystä ja päädyimme tässä vaiheessa seuraavanlaiseen ratkaisuun: tiedot viedään ensin metahakemistoon ja sieltä LDAP-hakemistoon ja aktiivihakemistoon. Winhasta valittiin identifioivaksi kentäksi opiskelijanumero. Opinnäytetyön tekijä sai uutta tietoa Winhasta. Kokouksessa mukana olivat Asmo Myllyaho, Juha Käenmäki, Marko Loukkaanhuhta ja Ari Sivula.

12.4.2005

Kokouksessa päätettiin, että henkilöstöhallinnasta tulisi henkilöstön primärilähde, sekä Winhasta ja Studentasta opiskelijoiden primärilähde. Kokouksessa mietimme myös hieman LDAP-hakemiston eri roolivaihtoehtoja, eli kenellä on hakemistossa oikeus tehdä ja mitä. Kokouksessa nousi myös esille mitä LDAP-hakemistoon ajettaisiin. LDAP-hakemistoon ajettaisiin mm. seuraavia tietoja: opiskelijan tiedot (etunimi, sukunimi, jne…), ryhmätunnus, suuntautumisvaihtoehto, opiskelijanumero, sähköpostiosoite ja henkilöllisyysturvatunnus. Puhuimme myös eri järjestelmistä, jota tullaan liittämään tulevaan keskitettyyn käyttäjähallintaan. Järjestelmät, jotka tulevat mukaan ovat muun muassa Moodle, WebCT, Linux/Unix ja aktiivihakemisto. Kokouksessa mukana olivat Asmo Myllyaho, Juha Käenmäki, Marko Loukkaanhuhta ja Ari Sivula.

9.5.2005

Kokouksessa päätettiin, että Seinäjoen koulutuskuntayhtymän korkeakoulukirjaston todennus voitaisiin hoitaa LDAP-hakemistosta. Salasanan vaihdosta päätettiin, että se laitettaisiin yhteen ja samaan käyttöliittymään. Tämä olisi myös kätevä myös silloin, kun henkilö olisi unohtanut salasanansa. Valmistuville ja poissaoleville kehitettäisiin eri säännöt, eli henkilön tunnus olisi disabloituna tietyn aikaa ja tämän jälkeen tunnus poistettaisiin järjestelmästä kokonaan

Mietimme myös mitä tietoja henkilöstöhallintojärjestelmästä tarvittaisiin LDAP-hakemistoon. Kokouksessa oli mukana laajempi ryhmä, eli Marko Loukkaanhuhta, Asmo Myllyaho, Juha Käenmäki, Jarkko Meronen, Seppo Salo, Hannele Husa, Jarmo Jaskari, Hannu Mansikka, Jukka Hokkanen ja Ari Sivula.

13.5.2005

Vierailimme Tampereen ammattikorkeakoulussa (TAMK) tutustumassa heidän järjestelmäänsä. Tampereen ammattikorkeakoulussa kyseistä järjestelmää on kehitetty jo viisi vuotta. TAMK:ssa on käytössä Sun Java Enterprise System keskitettyyn

käyttäjähallintaan. TAMK:ssa primäärilähteenä toimii Winha. Winhasta ajetaan myös kurssitiedot LDAP-hakemistoon. TAMK:ssa on rakennettu tämän ominaisuuden päälle monenlaisia sovelluksia. Sovelluksilla saadaan mm. kurssiryhmät haettua helposti ja suurien tiedostojen lähetys onnistuu tällä paremmin, koska ne eivät ole sähköpostin tukkeena. Kun TAMK:ssa alettiin ottaa järjestelmää käyttöön, niin vanhat tunnukset haettiin aktiivihakemistosta CSV-tiedostoksi ja SQL-hauilla haettiin Winhasta tiedot ja verrattiin niitä aktiivihakemiston tietoihin. TAMK:ssa meille esiteltiin myös tietohallinnon toimintaohjausjärjestelmä Efecteä. Tampereella mukana olivat Asmo Myllyaho, Juha Käenmäki, Marko Loukkaanhuhta, Jaakko Riihimaa, Hannele Husa, Hannu Mansikka, Seppo Salo, Jouni Vuorela, Jaakko Nuolikoski, Jarmo Jaskari ja Ari Sivula.

23.6.2005

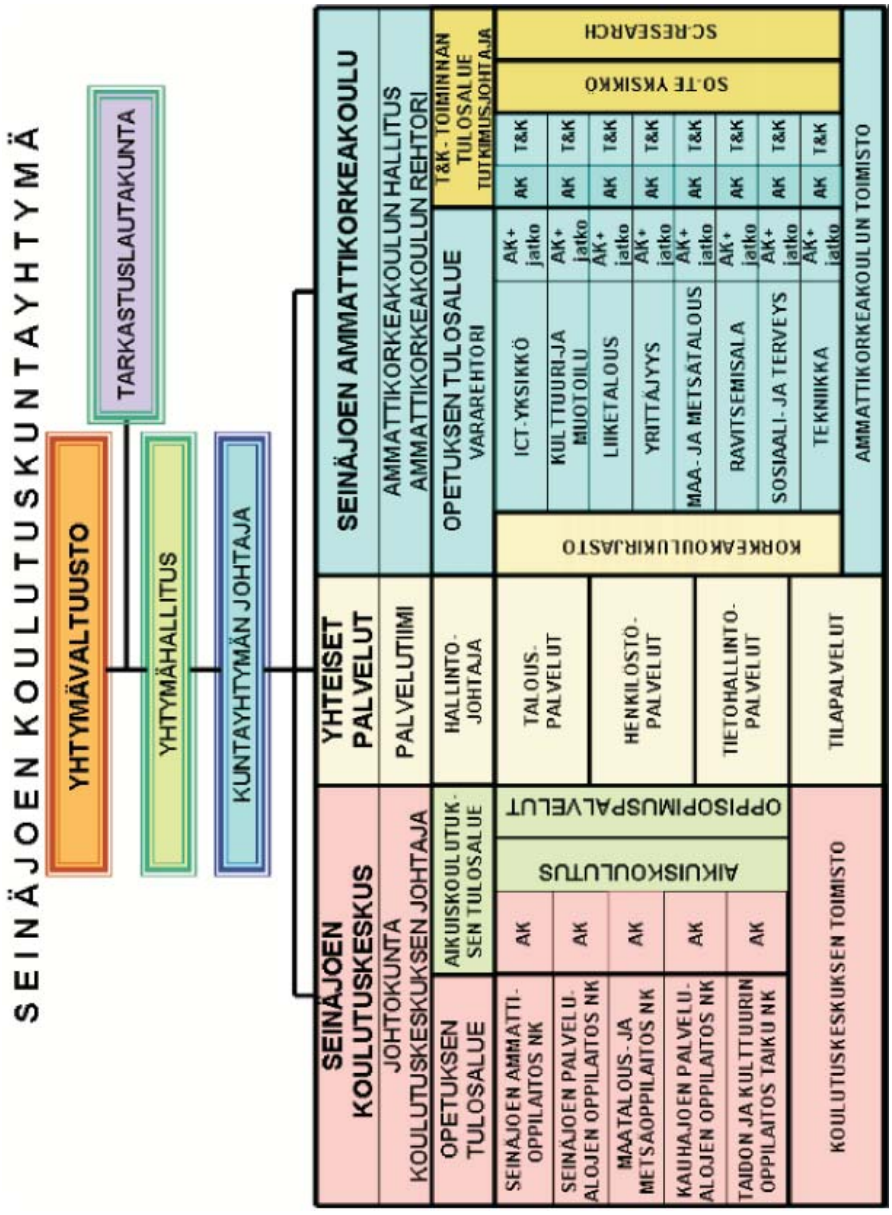
Kokouksessa päätettiin, että Winhasta ajetaan tiedot CSV-tiedostoksi ja siitä metahakemistoon. Mietimme myös olisiko olemassa valmistyökalua aktiivihakemiston ja Active Directory Application Moden välille. Kokouksessa myös päätös siitä, että kurssitiedot ajetaan koodeina LDAP-hakemistoon. Kokouksessa mukana Asmo Myllyaho, Marko Loukkaanhuhta, Seppo Salo ja Ari Sivula.

7.9.2005

FunetEduPerson -skeemaan päätettiin lisätä kaksi uutta attribuuttia. Attribuutit olivat luokkatunnus ja henkilön rooli (opiskelija vai henkilökuntaa). Uusi palvelin päätettiin hankkia Microsoft Identity Integration Server 2003:a varten. SQL Server 2000 päätettiin asentaa samalle palvelimille, kun MIIS 2003.

Winhasta tullessa tieto, että henkilö on valmistunut, niin tili on disabloituna 18 kuukautta LDAP-hakemistossa, 6 kuukautta aktiivihakemistossa ja poistuu tämän jälkeen. Kävi ilmi, että Studenta-opiskelijahallintojärjestelmästä ei löydy ollenkaan opiskelijanumeroa. Studentan identifioivaksi kentäksi valittiin studentasta löytyvä joukseva kenttänumero, joka on jokaisella opiskelijalla omansa. Kokouksessa mukana Asmo Myllyaho, Marko Loukkaanhuhta, Juha Käenmäki, Jarmo Jaskari, Veli-Matti Mäkelä ja Ari Sivula.

Liite 2. Seinäjoen koulutuskuntayhtymän organisaatiokaavio. (Lähde: Myllyaho, A., sähköpostikeskustelu 7.10.2005)



Liite 3. FunetEduPerson-skeema X.500-muodossa.

```
# funetEduPerson LDAP-schema v. 1.0/16.6.2003
# 18.11.2003: korjattu attributetype -> attributetypes:
# ja objectclass -> objectclasses:
#
# Käännetty X.500-muotoon 12.8.2005 (Ari Sivula)
# Lisätty 2 attribuuttia: funetEduPersonAffiliation ja funetEduPersonGroup 15.9.2005 (Ari Sivula)
#
# Skeeman vienti ADAM:iin komennolla:
# ldifde -i -f funetEduPerson_1_0.ldf -s server:port -k -j . -c ”CN=Schema,CN=Configuration,DC=X”
#schemaNamingContext
#
#_____________________________________________________________________________
#
# attributes:
#
#
# funetEduPersonIdentityCode
# Finnish Identity Code
# alphanumeric
# single-valued

dn: CN=funetEduPersonIdentityCode,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonIdentityCode
attributeID: 1.3.6.1.4.1.16161.1.1.1
attributeSyntax: 2.5.5.12
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonIdentityCode
adminDescription: Finnish Identity Code, earlier social security number
description: Finnish Identity Code, earlier social security number
oMSyntax: 64
lDAPDisplayName: funetEduPersonIdentityCode
systemOnly: FALSE


# funetEduPersonDateOfBirth
# specifies a person’s date of birth
# alphanumeric
# single-valued

dn: CN=funetEduPersonDateOfBirth,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonDateOfBirth
attributeID: 1.3.6.1.4.1.16161.1.1.2
attributeSyntax: 2.5.5.12
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonDateOfBirth
adminDescription: date of birth
description: date of birth
oMSyntax: 64
lDAPDisplayName: funetEduPersonDateOfBirth
```

```
systemOnly: FALSE


# funetEduPersonTargetDegreeUniversity
# specifies the person´s target degree in a University
# numeric
# multi-valued
# values: Central Statistical Office of Finland:
# http://www.tilastokeskus.fi/tk/he/tiedonkeruu_oppilaitokset/yliopistot_tutkinnot.xls

dn: CN=funetEduPersonTargetDegreeUniversity,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonTargetDegreeUniversity
attributeID: 1.3.6.1.4.1.16161.1.1.3
attributeSyntax: 2.5.5.6
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonTargetDegreeUniversity
adminDescription: university target degree
description: university target degree
oMSyntax: 18
lDAPDisplayName: funetEduPersonTargetDegreeUniversity
systemOnly: FALSE


# funetEduPersonTargetDegreePolytech
# specifies the person´s target degree in a Polytechnic
# numeric
# multi-valued
# values: Central Statistical Office of Finland:
# http://www.tilastokeskus.fi/tk/he/tiedonkeruu_oppilaitokset/amk_tutkintokoodit.xls

dn: CN=funetEduPersonTargetDegreePolytech,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonTargetDegreePolytech
attributeID: 1.3.6.1.4.1.16161.1.1.4
attributeSyntax: 2.5.5.6
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonTargetDegreePolytech
adminDescription: polytechnical target degree
description: polytechnical target degree
oMSyntax: 18
lDAPDisplayName: funetEduPersonTargetDegreePolytech
systemOnly: FALSE


# funetEduPersonEducationalProgramUniv
# specifies a student´s educational program (using Stat classification, koulutusohjelma)
# numeric
# multi-valued
# values: http://www.tilastokeskus.fi/tk/he/tiedonkeruu_oppilaitokset/yliopistot_kokoodit.xls

dn: CN=funetEduPersonEducationalProgramUniv,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
```

```
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonEducationalProgramUniv
attributeID: 1.3.6.1.4.1.16161.1.1.5
attributeSyntax: 2.5.5.6
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonEducationalProgramUniv
adminDescription: educational program
description: educational program
oMSyntax: 18
lDAPDisplayName: funetEduPersonEducationalProgramUniv
systemOnly: FALSE


# funetEduPersonEducationalProgramPolytech
# specifies a student´s educational program (using Stat classification, koulutusohjelma)
# numeric
# multi-valued
# values: http://www.tilastokeskus.fi/tk/he/tiedonkeruu_oppilaitokset/amk_kokoodit.xls

dn: CN=funetEduPersonEducationalProgramPolytech,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonEducationalProgramPolytech
attributeID: 1.3.6.1.4.1.16161.1.1.6
attributeSyntax: 2.5.5.6
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonEducationalProgramPolytech
adminDescription: educational program
description: educational program
oMSyntax: 18
lDAPDisplayName: funetEduPersonEducationalProgramPolytech
systemOnly: FALSE


# funetEduPersonOrientationAlternPolytech
# specifies the orientation alternative of a student (using Stat classification, suunt.vaihtoehto)
# numeric
# multi-valued
# values: http://www.tilastokeskus.fi/tk/he/tiedonkeruu_oppilaitokset/amk_svkoodit.xls

dn: CN=funetEduPersonOrientationAlternPolytech,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonOrientationAlternPolytech
attributeID: 1.3.6.1.4.1.16161.1.1.7
attributeSyntax: 2.5.5.6
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonOrientationAlternPolytech
adminDescription: orientation alternative
description: orientation alternative
oMSyntax: 18
lDAPDisplayName: funetEduPersonOrientationAlternPolytech
systemOnly: FALSE
```

```
# funetEduPersonMajorUniv
# specifies the main academic subjects of a student (using Stat classification, pääaine), only for a
# university student
# numeric
# multi-valued
# values: http://www.tilastokeskus.fi/tk/he/tiedonkeruu_oppilaitokset/yliopistot_pakoodit.xls

dn: CN=funetEduPersonMajorUniv,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonMajorUniv
attributeID: 1.3.6.1.4.1.16161.1.1.8
attributeSyntax: 2.5.5.6
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonMajorUniv
adminDescription: main subject(s)
description: main subject(s)
oMSyntax: 18
lDAPDisplayName: funetEduPersonMajorUniv
systemOnly: FALSE


# funetEduPersonHomeOrganization
# specifies a person´s home university = domain name of the Organization
# alphanumeric
# single-valued
# values: domain name of a home organization (e.g. tut.fi)

dn: CN=funetEduPersonHomeOrganization,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonHomeOrganization
attributeID: 1.3.6.1.4.1.16161.1.1.9
attributeSyntax: 2.5.5.12
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonHomeOrganization
adminDescription: domain name of the person´s home organization
description: domain name of the person´s home organization
oMSyntax: 64
lDAPDisplayName: funetEduPersonHomeOrganization
systemOnly: FALSE


# funetEduPersonStudentID OID 1.3.6.1.4.1.16161.1.1.10
# specifies a person´s student number or code
# alphanumeric
# multi-valued
# values:

dn: CN=funetEduPersonStudentID,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonStudentID
attributeID: 1.3.6.1.4.1.16161.1.1.10
```

```
attributeSyntax: 2.5.5.12
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonStudentID
adminDescription: student number or code
description: student number or code
oMSyntax: 64
lDAPDisplayName: funetEduPersonStudentID
systemOnly: FALSE

# funetEduPersonAffiliation OID 1.3.6.1.4.1.16161.1.1.11
# specifies a person's affiliation
# numeric
# single-valued
# values: 1 or 2, 1 = student and 2 = staff

dn: CN=funetEduPersonAffiliation,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonAffiliation
attributeID: 1.3.6.1.4.1.16161.1.1.11
attributeSyntax: 2.5.5.6
isSingleValued: TRUE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonAffiliation
adminDescription: person's affiliation
description: person's affiliation
oMSyntax: 18
lDAPDisplayName: funetEduPersonAffiliation
systemOnly: FALSE

# funetEduPersonGroup OID 1.3.6.1.4.1.16161.1.1.12
# specifies a person´s students group
# alphanumeric
# multi-valued
# values:

dn: CN=funetEduPersonGroup,CN=Schema,CN=Configuration,DC=X
changetype: ntdsschemaadd
objectClass: top
objectClass: attributeSchema
cn: funetEduPersonGroup
attributeID: 1.3.6.1.4.1.16161.1.1.12
attributeSyntax: 2.5.5.12
isSingleValued: FALSE
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPersonGroup
adminDescription: student group
description: student group
oMSyntax: 64
lDAPDisplayName: funetEduPersonGroup
systemOnly: FALSE

#-------------------------------------------------------------
# funetEduPerson Object Class:
#

dn: CN=funetEduPerson,CN=Schema,CN=Configuration,DC=X
```

```
changetype: ntdsschemaadd
objectClass: top
objectClass: classSchema
cn: funetEduPerson
subClassOf: top
governsID: 1.3.6.1.4.1.16161.1.1
mustContain: funetEduPersonHomeOrganization
mayContain: funetEduPersonIdentityCode
mayContain: funetEduPersonDateOfBirth
mayContain: funetEduPersonTargetDegreePolytech
mayContain: funetEduPersonTargetDegreeUniversity
mayContain: funetEduPersonEducationalProgramUniv
mayContain: funetEduPersonEducationalProgramPolytech
mayContain: funetEduPersonOrientationAlternPolytech
mayContain: funetEduPersonMajorUniv
mayContain: funetEduPersonStudentID
mayContain: funetEduPersonAffiliation
mayContain: funetEduPersonGroup
rDNAttID: cn
showInAdvancedViewOnly: TRUE
adminDisplayName: funetEduPerson
adminDescription: funetEduPerson
objectClassCategory: 0
lDAPDisplayName: funetEduPerson
name: funetEduPerson
systemOnly: FALSE
systemPossSuperiors: domainDNS
systemPossSuperiors: organizationalUnit
systemPossSuperiors: container
systemAuxiliaryClass: securityPrincipal
systemAuxiliaryClass: msDS-BindableObject
defaultSecurityDescriptor:
 D:S:
defaultHidingValue: TRUE
defaultObjectCategory: CN=funetEduPerson,CN=Schema,CN=Configuration,DC=X

dn:
changetype: modify
add: schemaUpdateNow
schemaUpdateNow: 1
-
# end
```

Liite 4. Winhan CSV-tiedoston attribuuttilistaus.
(Lähde: Winhapro 4.5 LDAP, 2005)

funetEduPersonStudentId

Opiskelijan roolitunnus / henkilön tunnus.

winhaPersonStudentOtherIdList

Opiskelijan muut aktiiviset roolit, kaksoispiste väliin. Näiden roolien voimassaoleva läsnäolo ei saa olla LDAP-kielelle käännettynä VP tai ER.

Henkilöllä tyhjä.
Huom! Henkilö kirjoitetaan tiedostoon vain kerran kotitoimipisteen mukaan, vaikka hänellä olisi rooleja eri toimipisteissä.

winhaPersonSatu

Tyhjä.

winhaAttendsCourse

Opiskelijan osallistuminen toteutuksille (suoritushistorian mukaan). Tähän luetellaan kaikki ne meneillään olevat toteutukset (opinto/opityyp/toteutus, esim. XYZ0045/OJ/3), joissa HOPSin opinnon tilanne on H, P tai A eikä to-teutusta ole peruutettu.

Henkilön toteutukset. Tähän luetellaan kaikki ne meneillään olevat toteutuk-set (opinto/opityyp/toteutus, esim. XYZ0045/OJ/3), joissa henkilö on missä tahansa roolissa (opettaja, luennoitsija, tms.) eikä toteutusta ole peruutettu.

winhaEnteringGroup

Opiskelijan saapumisryhmä.
Henkilöllä tyhjä.

winhaGroup

Opiskelijan hallinnolliset ryhmät, kaksoispiste väliin.
Henkilön henkilöryhmät, kaksoispiste väliin.

winhaAttendanceCode

Opiskelijan voimassa oleva läsnäolokoodi LDAP-kielelle käännettynä.
Henkilön voimassa oleva läsnäolokoodi LDAP-kielelle käännettynä.

funetEduPersonEducationalProgramPolytech

Opiskelijan koulutusohjelman mukainen TK-koodi.
Henkilöllä tyhjä.

winhaOptionPreference

Opiskelijan suuntavaihtoehdon TK-koodi, kaksoistutkinnoilla molempien suuntien TK-koodit, kaksoispiste väliin.
Henkilöllä tyhjä.

winhaInternetPermission

Opiskelijan luovutuslupa internetiin, 1 = kyllä, 0 = ei

winhaIntranetPermission

Opiskelijan luovutuslupa intranetiin, tyhjä (kenttää ei ole Winhassa)

winhaEducPermission

Opiskelijan luovutuslupa koulutustiedotukseen , 1 = kyllä, 0 = ei

winhaOrganisationUnit

Opiskelijan saapumisryhmän toimipiste.
Henkilön kotitoimipiste.

winhaMarketingPermission

Opiskelijan luovutuslupa markkinointiin, 1 = kyllä, 0 = ei

sn (surname)

Opiskelijan sukunimi
Henkilön sukunimi

givenName

Opiskelijan etunimet
Henkilön etunimet

telephoneNumber

Opiskelijan puhelinnumero
Henkilön puhelinnumero

street

Opiskelijan asuinosoite (erotinmerkkinä ”>>”)
Henkilön kotiosoite (erotinmerkkinä ”>>”)

postalAddress

Opiskelijan asuinosoitteen postitoimipaikka
Henkilön kotiosoitteen postitoimipaikka

postalCode

Opiskelijan asuinosoitteen postinumero
Henkilön kotiosoitteen postinumero

preferredLanguage

Opiskelijan käyttökieli
Henkilön käyttökieli

mail

Opiskelijan sähköpostiosoite
Henkilön kotitoimipisteen sähköpostiosoite

mobile

Opiskelijan asuinosoitteen 2. puhelin
Henkilön kotitoimipisteen työpuhelin

eduPersonAffiliation

vakio
1 = opiskelija (student)
2 = henkilö (staff)

Toiminimike

Opiskelijalla tyhjä.
Henkilön toiminimike, käännettynä LDAP-kielelle.

Opintoala

Opiskelijalla tyhjä.
Henkilön opintoala.

Kutsumanimi

Opiskelijalla kutsumanimi.
Henkilöllä tyhjä.

Koulutus

Opiskelijan koulutustyyppi, käännettynä LDAP-kielelle.
Henkilöllä tyhjä.

Liite 5. Seinäjoen ammattikorkeakoulun yksikkötunnukset ja Winhasta tulevat yksikkötunnukset.
(Lähde: Myllyaho, A., sähköpostikeskustelu 7.10.2005 & Käenmäki, J., haastattelu 11.10.2005)

Yksikkö-tunnus	**Yksikön nimi**	**winhaOrganisationUnit**
A	Maaseutualan yksikkö/Ilmajoki	KMAASEUTU
B	Metsäalan yksikkö/Ähtäri	KMETSÄ
C	Tekniikan yksikkö/Seinäjoki	KTEKNIIKKA
D	Liiketalouden yksikkö/Seinäjoki	KKAUPPA
E	Yrittäjyyden yksikkö/Kauhava	KYRITTÄJÄ
F	Sosiaali- ja terveysalan yksikkö, Kampusalueen toimipiste/Seinäjoki	KSOSIAALI
G	Sosiaali- ja terveysalan yksikkö, Koskenalantien toimipiste/Seinäjoki	KTERVEYS
H	Kulttuurialan ja muotoilun yksikkö/ Jurva, Seinäjoki	KKULTTUURI KKULTKONS KKULTKUTU (L)
I	Ravitsemisalan yksikkö/Kauhajoki	KRAVITSEMI
K	Korkeakoulukirjasto	KORKEAKK KIRJASTO
L	Informaatio- ja kommunikaatio-teknologian yksikkö/Seinäjoki	KICT
X	Seitek	SEITEK
Z	SC-Research	KSCR

Liite 6. ADAM:in Metaverse Rules Extensionin Provision-metodi.

```
void IMVSynchronization.Provision (MVEntry mventry)
{

    string Agent_Contributing = null;
    string Agent_Target = null;
    string DN = null;

    // Määritellään MA:t
    Agent_Contributing = @"Winha_MA"; // Winhan Management Agent (lähde).
    Agent_Target = @"ADAM_MA"; // Active Directory Application Moden Management Agent (kohde).

    try
    {

        this.Log(@"Provisioidaan... ");

        this.Log(string.Concat("Objektin Tyyppi: ",mventry.ObjectType));

        // Tarkistetaan, että objektin tyyppi on Person
        switch(mventry.ObjectType)
        {

            case @"person":
                break;
            default:
                return;
    }

    this.Log(string.Concat("Jakelu Agentti: ",mventry[@"cn"].LastContributingMA.Name));

    // Tarkistetaan, että Winhan Management Agent on viimeksi provisioinut objekteja ADAM:iin.
    if(string.Compare(mventry[@"cn"].LastContributingMA.Name,Agent_Contributing,true)!=0)
    {
        return;
    }

    // Määritetään Management Agent, jolle syötetään tietoa.
    ConnectedMA Agent = mventry.ConnectedMAs[Agent_Target];
    this.Log(string.Concat(@"Kohde Agentti: ",Agent.Name));
    int Connectors = Agent.Connectors.Count;

    string yksikko = null;
    string opiskelijat = null;

    // Tehdään OU objektille winhaOrganisationUnitista riippuen.
    if (mventry["winhaOrganisationUnit"].Value.Equals("KMAASEUTU"))
    {
        yksikko = ",OU=Maa-Metsa";
        opiskelijat = ",OU=Opiskelijat,OU=Ilmajoki";
    }

    if (mventry["winhaOrganisationUnit"].Value.Equals("KMETSÄ"))
    {
        yksikko = ",OU=Maa-Metsa";
        opiskelijat = ",OU=Opiskelijat,OU=Opetus,OU=Tuomarniemi";
    }

    if (mventry["winhaOrganisationUnit"].Value.Equals("KTEKNIIKKA"))
    {
        yksikko = ",OU=Tekniikka";
        opiskelijat = ",OU=Opiskelijat";
    }

    if (mventry["winhaOrganisationUnit"].Value.Equals("KKAUPPA"))
    {
        yksikko = ",OU=Liiketalous";
        opiskelijat = ",OU=Opiskelijat";
    }
```

```
if (mventry["winhaOrganisationUnit"].Value.Equals("KYRITTÄJÄ"))
{
    yksikko = ",OU=Yrittajyys";
    opiskelijat = ",OU=eOpiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KSOSIAALI"))
{
    yksikko = ",OU=Sosiaali-Terveys";
    opiskelijat = ",OU=Opiskelijat,OU=Kampusalue";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KTERVEYS"))
{
    yksikko = ",OU=Sosiaali-Terveys";
    opiskelijat = ",OU=Opiskelijat,OU=Koskenala";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKULTTUURI"))
{
    yksikko = ",OU=Kulttuuri-muotoilu";
    opiskelijat = ",OU=AMK,OU=Opiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKULTKONS"))
{
    yksikko = ",OU=Kulttuuri-muotoilu";
    opiskelijat = ",OU=Opiskelijat,OU=konservaattorit";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KKULTKUTU"))
{
    yksikko = ",OU=ICT";
    opiskelijat = ",OU=Kulttuuri,OU=Opiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KRAVITSEMI"))
{
    yksikko = ",OU=Ravitsemisala";
    opiskelijat = ",OU=AmkOpiskelijat";
}

if (mventry["winhaOrganisationUnit"].Value.Equals("KICT"))
{
    yksikko = ",OU=ICT";

    // Katsotaan, onko henkilö TIKO vai TITE ryhmästä.
    for (int i = 0; i < mventry["winhaEnteringGroup"].Value.Length; i++)
    {
        try
        {
            if (mventry["winhaEnteringGroup"].Value.Substring(i, 4).Equals("TIKO"))
            {
                opiskelijat = ",OU=TIKO,OU=Opiskelijat";
                break;
            }

            if (mventry["winhaEnteringGroup"].Value.Substring(i, 4).Equals("TITE"))
            {
                opiskelijat = ",OU=TITE,OU=Opiskelijat";
                break;
            }

            else
            {
                opiskelijat = ",OU=Muut,OU=Opiskelijat";
            }
        }

        catch (Exception)
        {
            continue;
        }
    }
}
```

```
try
{
    if (mventry["winhaAttendanceCode"].Value.Equals("VP"))
    {
        // Henkilön valmistuessa, tili siirretään.
        DN = "CN="+mventry["sn"].Value+"\\, "+mventry["cn"].Value+","+"OU=Valmistuneet,OU=Disabloidut"+
             opiskelijat+yksikko+",OU=SeAMK,O=SeAMK,C=fi";
    }

    else if (mventry["winhaAttendanceCode"].Value.Equals("ER"))
    {
        // Henkilön erotessa, tili siirretään.
        DN = "CN="+mventry["sn"].Value+"\\, "+mventry["cn"].Value+","+"OU=Eronneet,OU=Disabloidut"+
             opiskelijat+yksikko+",OU=SeAMK,O=SeAMK,C=fi";
    }

    else
    {
        // Luodaan DN-string normaalille objektille.
        DN = "CN="+mventry["sn"].Value+"\\, "+mventry["cn"].Value+","+"OU="+mventry["winhaEnteringGroup"].Value+
             opiskelijat+yksikko+",OU=SeAMK,O=SeAMK,C=fi";
    }

}
catch(Exception cException)
{
    this.Log(string.Concat(@"Virhe: ",cException.Message));
    return;
}
this.Log(string.Concat(@"Provisioidaan ",mventry[@"cn"].Value));

// Määritetään Distinguished Name objektille DN-stringistä.
ReferenceValue DistinguishedName = Agent.CreateDN(DN);

this.Log(string.Concat(@"DN: ",DistinguishedName.ToString()));

CSEntry csentry = null;

switch(Connectors)
{
    case 0:
        this.Log(@"Luodaan Uusi Connectori.");

        string sObjectType = null;

    // Tarkistetaan objektityyppi. Metaversestä tulevan objektin tyyppi pitää olla Person.
    switch(mventry.ObjectType)
    {
        case @"person":
            // Määritetään hakemiston objektityypiksi funetEduPerson.
            sObjectType = "funetEduPerson";
            break;

        default:
            throw new UnexpectedDataException("Odottamaton Objektin Tyyppi.");

    }
        // Tallennetaan uusi connectori ADAM:in Connector Spaceen.
        csentry = Agent.Connectors.StartNewConnector(sObjectType);
        csentry.DN = DistinguishedName; // Määritetään Distinguished Name objektille.
        csentry["unicodepwd"].Values.Add("P@ssW0rd"); // Ensimmäinen salasana käyttäjälle.
        csentry["pwdLastSet"].Value = "0"; // Asetetaan pwdLastSet 0, eli salasanaa ei ole ikinä vaihdettu.
        csentry.CommitNewConnector(); // Luodaan uusi Connectori.
        break;

    case 1:
        // Jos objekti on jo olemassa, niin muokataan sitä.
        this.Log(@"Muokataan Olemassa Olevaa Connectoria.");
        csentry = Agent.Connectors.ByIndex[0];
        csentry.DN = DistinguishedName;
        break;

    default:
        throw new UnexpectedDataException("Moninkertainen Connectori Management Agentilla!");
}
}
```

```
        catch (MissingParentObjectException)
        {

                // Kirjoitetaan lokiin puuttuvien OU-haarojen DN:t OUCreatoria varten.
                Microsoft.MetadirectoryServices.Logging.Logging.SetupLogFile("MissingOUs.txt", 10);
                Microsoft.MetadirectoryServices.Logging.Logging.Log(DN, false, 10);
                // Vaihdetaan normaali loki takaisin.
                Microsoft.MetadirectoryServices.Logging.Logging.SetupLogFile("Provisioning.log", 10);
        }


        catch(Exception cException)
        {
            // Virheet kirjoitetaan lokiin.
            this.Log(((cException.Message == null)||(cException.Message == string.Empty))?
                        @"Tuntematon virhe.":string.Concat("Virhe: ",cException.Message));
            throw cException;
        0}
}
```

Liite 7. MIIS 2003:n eri skriptit.

RunMA.vbs

```
Option Explicit

On Error Resume Next

Const PktPrivacy = 6
Dim ErrorLevel
Dim Arguments
Dim ManagementAgentName
Dim RunProfileName
Dim Service
Dim ManagementAgent
Dim Status

ErrorLevel = 1

Set Arguments = WScript.Arguments.Named

ManagementAgentName = Arguments.Item("MA")

RunProfileName = Arguments.Item("RunProfile")

Set Service = GetObject("winmgmts:{authenticationLevel=PktPrivacy}!root\MicrosoftIdentityIntegrationServer")

Set ManagementAgent = Service.Get("MIIS_ManagementAgent.Name='" & ManagementAgentName & "'")

Status = ManagementAgent.Execute(RunProfileName)

WScript.Echo ManagementAgentName & " " & RunProfileName & " Result: " & Status

If Status = "success" then

  ErrorLevel = 0

End If

Sub ErrorHandler (ErrorMessage)
 WScript.Echo ErrorMessage
 WScript.Quit(1)
End Sub
```

AD_Import.vbs

```
Const PktPrivacy = 6
rem Const wbemAuthenticationLevelPkt = 6
Set Locator = CreateObject("WbemScripting.SWbemLocator")
rem
rem Credentials must only be specified when Microsoft Identity Integration Server is on remote system.
rem
rem Locator.Security_.AuthenticationLevel = wbemAuthenticationLevelPkt
rem Set Service = Locator.ConnectServer("MyServer", "root/MicrosoftIdentityIntegrationServer")
rem Set Service = Locator.ConnectServer("MyServer", "root/MicrosoftIdentityIntegrationServer", "Domain\Me", "MyPassword")
rem
Set Service = GetObject("winmgmts:{authenticationLevel=PktPrivacy}!root/MicrosoftIdentityIntegrationServer")
Set MASet   = Service.ExecQuery("select * from MIIS_ManagementAgent where Guid = '{D33FA63F-B5E2-4480-B346-27D66AAB718A}'")

for each MA in MASet
   WScript.Echo "Running " + MA.name + ".Execute(""Import"")..."
   WScript.Echo "Run completed with result: " + MA.Execute("Import")
next
```

ADAM_Import.vbs

```
Const PktPrivacy = 6
rem Const wbemAuthenticationLevelPkt = 6
Set Locator = CreateObject("WbemScripting.SWbemLocator")
rem
rem Credentials must only be specified when Microsoft Identity Integration Server is on remote system.
rem
rem Locator.Security_.AuthenticationLevel = wbemAuthenticationLevelPkt
rem Set Service = Locator.ConnectServer("MyServer", "root/MicrosoftIdentityIntegrationServer")
rem Set Service = Locator.ConnectServer("MyServer", "root/MicrosoftIdentityIntegrationServer", "Domain\Me", "MyPassword")
rem
Set Service = GetObject("winmgmts:{authenticationLevel=PktPrivacy}!root/MicrosoftIdentityIntegrationServer")
Set MASet   = Service.ExecQuery("select * from MIIS_ManagementAgent where Guid = '{A8177844-DD6E-4C0F-84D7-2DEF17246D55}'")

for each MA in MASet
   WScript.Echo "Running " + MA.name + ".Execute(""Import"")..."
   WScript.Echo "Run completed with result: " + MA.Execute("Import")
next
```

Winha_FullSynchronization.vbs

```
Const PktPrivacy = 6
rem Const wbemAuthenticationLevelPkt = 6
Set Locator = CreateObject("WbemScripting.SWbemLocator")
rem
rem Credentials must only be specified when Microsoft Identity Integration Server is on remote system.
rem
rem Locator.Security_.AuthenticationLevel = wbemAuthenticationLevelPkt
rem Set Service = Locator.ConnectServer("MyServer", "root/MicrosoftIdentityIntegrationServer")
rem Set Service = Locator.ConnectServer("MyServer", "root/MicrosoftIdentityIntegrationServer", "Domain\Me", "MyPassword")
rem
Set Service = GetObject("winmgmts:{authenticationLevel=PktPrivacy}!root/MicrosoftIdentityIntegrationServer")
Set MASet   = Service.ExecQuery("select * from MIIS_ManagementAgent where Guid = '{0001AA9D-14C1-48C0-8A72-9F5812DF1B5B}'")

for each MA in MASet
   WScript.Echo "Running " + MA.name + ".Execute(""Full Sync"")..."
   WScript.Echo "Run completed with result: " + MA.Execute("Full Sync")
next
```

SEINÄJOEN AMMATTIKORKEAKOULUN JULKAISUSARJA

A. TUTKIMUKSIA

1. Timo Toikko. Sosiaalityön amerikkalainen oppi. Yhdysvaltalaisen caseworkin kehitys ja sen yhteys suomalaiseen tapauskohtaiseen sosiaalityöhön. 2001.

2. Jouni Björkman. Risk Assessment Methods in System Approach to Fire Safety. 2005.

B. RAPORTTEJA JA SELVITYKSIÄ

1. Seinäjoen ammattikorkeakoulusta soveltavan osaamisen korkeakoulu – tutkimus- ja kehitystoiminnan ohjelma. 1998.

2. Elina Varamäki - Ritva Lintilä - Taru Hautala - Eija Taipalus. Pk-yritysten ja ammattikorkeakoulun yhteinen tulevaisuus: prosessin kuvaus, tuotokset ja toimintaehdotukset. 1998.

3. Elina Varamäki - Tarja Heikkilä - Eija Taipalus. Ammattikorkeakoulusta työelämään: Seinäjoen ammattikorkeakoulusta 1996-1997 valmistuneiden sijoittuminen. 1999.

4. Petri Kahila. Tietoteollisen koulutuksen tilanne- ja tarveselvitys Seinäjoen ammattikorkeakoulussa: väliraportti. 1999.

5. Elina Varamäki. Pk-yritysten tuleva elinkaari - säilyykö Etelä-Pohjanmaa yrittäjämaakuntana? 1999.

6. Seinäjoen ammattikorkeakoulun laatujärjestelmän auditointi 1998–1999. Itsearviointiraportti ja keskeiset tulokset. 2000.

7. Heikki Ylihärsilä. Puurakentaminen rakennusinsinöörien koulutuksessa. 2000.

8. Juha Ruuska. Kulttuuri- ja sisältötuotannon koulutusselvitys. 2000.

9. Seinäjoen ammattikorkeakoulusta soveltavan osaamisen korkeakoulu. Tutkimus- ja kehitystoiminnan ohjelma 2001. 2001.

10. Minna Kivipelto (toim.). Sosionomin asiantuntijuus. Esimerkkejä kriminaalihuolto-, vankila- ja projektityöstä. 2001.

11. Elina Varamäki - Tarja Heikkilä - Eija Taipalus. Ammattikorkeakoulusta työelämään. Seinäjoen ammattikorkeakoulusta 1998–2000 valmistuneiden sijoittuminen. 2002.

12. Varmola T., Kitinoja H. & Peltola A. (ed.) Quality and new challenges of higher education. International Conference 25.-26. September, 2002. Seinäjoki Finland. Proceedings. 2002.

13. Susanna Tauriainen & Arja Ala-Kauppila. Kivennäisaineet kasvavien nautojen ruokinnassa. 2003.

14. Päivi Laitinen & Sanna Välisaari. Staphylococcus aureus -bakteerien aiheuttaman utaretulehduksen ennaltaehkäisy ja hoito lypsykarja-tiloilla. 2003.

15. Riikka Ahmaniemi & Marjut Setälä. Seinäjoen ammattikorkeakoulu – Alueellinen kehittäjä, toimija ja näkijä. 2003.

16. Hannu Saari & Mika Oijennus. Toiminnanohjaus kehityskohteena pk-yrityksessä. 2004.

17. Leena Niemi. Sosiaalisen tarkastelua. 2004.

18. Marko Järvenpää (toim.) Muutoksen kärjessä. Kalevi Karjanlahti 60 vuotta. 2004.

19. Suvi Torkki (toim.). Kohti käyttäjäkeskeistä muotoilua. Muotoilija-koulutuksen painotuksia SeAMK:ssa. 2005.

20. Timo Toikko (toim.). Sosiaalialan kehittämistyön lähtökohta. 2005.

21. Elina Varamäki & Tarja Heikkilä & Eija Taipalus. Ammattikorkea-koulusta työelämään. Seinäjoen ammattikorkeakoulusta v. 2001–2003 valmistuneiden sijoittuminen opiskelun jälkeen. 2005.

22. Tuija Pitkäkoski, Sari Pajuniemi & Hanne Vuorenmaa (ed.). Food Choices and Healthy Eating. Focusing on Vegetables, Fruits and Berries. International Conference *September* 2nd – 3rd 2005. Kauhajoki, Finland.Proceedings. 2005.

23. Katariina Perttula. Kokemuksellinen hyvinvointi Seinäjoen kolmella asuinalueella. Raportti pilottihankkeen tuloksista. 2005.

24. Mervi Lehtola. Alueellinen hyvinvointitiedon malli – asiantuntijat puhujina. Hankkeen loppuraportti. 2005.

25. Timo Suutari, Kari Salo & Sami Kurki. Seinäjoen Teknologia- ja innovaatiokeskus Frami vuorovaikutusta ja innovatiivisuutta edistävänä ympäristönä. 2006

C. OPPIMATERIAALEJA

1. Ville-Pekka Mäkeläinen. Basics of business to business marketing. 1999.

2. Lea Knuuttila. Mihin työohjausta tarvitaan? Oppimateriaalia sosiaali-alan opiskelijoiden työnohjauskurssille. 2001.

3. Mirva Kuni & Petteri Männistö & Markus Välimaa. Leikkauspelot ja niiden hoitaminen. 2002.

D. OPINNÄYTTEITÄ

1. Hanna Halmesmäki – Merja Halmesmäki. Työvoiman osaamistarve-kartoitus Etelä-Pohjanmaan metalli- ja puualan yrityksissä. 1999.

2. Tiina Kankaanpää – Maija Luoma-aho – Heli Sinisalo. Kymmenen metrin kävelytestin suoritusohjeet CD-rom levyllä: aivoverenkierto-häiriöön sairastuneen kävelyn mittaaminen. 2000.

3. Laura Elo. Arvojen rooli yritysmaailmassa. 2001.

4. Nina Anttila. Päälle käyvää – vaatemallisto ikääntyvälle naiselle. 2002.

5. Jaana Jeminen. Matkalla muotoiluyrittäjyyteen. 2002.

6. Päivi Akkanen. Lypsääkö meillä tulevaisuudessa robotti? 2002.

7. Johanna Kivioja. E-learningin alkutaival ja tulevaisuus Suomessa. 2002.

8. Heli Kuntola – Hannele Raukola. Naisen kokemuksia minäkuvan muuttumisesta rinnanpoistoleikkauksen jälkeen. 2003.

9. Jenni Pietarila. Meno-paluu –lauluillan tuottaminen. Produktion tuottajan käsikirja. 2003.

10. Johanna Hautamäki. Asiantuntijapalvelun tuotteistaminen case: ´Avaimet markkinointiin, kehittyvän yrityksen asiakasohjelma -pilotti projekti´. 2003.

11. Sanna-Mari Petäjistö. Teollinen tuotemuotoiluprosessi – Sohvapöydän ja sen oheistuotteiden suunnittelu. 2004.

12. Susanna Patrikainen. Nuorekkaita asukokonaisuuksia Mode LaRose Oy:lle. Vaatemallien suunnittelu teolliseen mallistoon. 2004.

13. Tanja Rajala. Suonikohjuleikkaukseen tulevan potilaan ja hänen perheensä ohjaus päiväkirurgisessa yksikössä. 2004.

14. Marjo Lapiolahti. Maksuvalmiuslaskelmien toteutuminen sukupolven-vaihdostiloilla. 2004.

15. Marjo Taittonen. Tutkimusmatka syrjäytymisen maailmaan. 2004.

16. Minna Hakala. Maidon koostumus ja laatutekijät. 2004.

17. Anne Uusitalo. Tuomarniemen ympäristöohjelma. 2004.

18. Maarit Hoffrén. Vaihtelua kasviksilla. Kasvisruokalistan kehittäminen opiskelijaravintola Risettiin. 2004.

19. Sami Karppinen. Tuomarniemen hengessä. Arkeista antologiaksi. 2005.

20. Elina Syrjänen – Anne-Mari Uschanoff. Messut – ideasta toimintaan. Messutoteutus osana yrityksen markkinointiviestintää. 2005.

SEINÄJOEN
AMMATTIKORKEAKOULU
SEINÄJOEN KORKEAKOULUKIRJASTO
Keskuskatu 34, 60100 Seinäjoki
Puh. 020 124 5040, fax 020 124 5041
E-mail seamk.kirjasto@seamk.fi

ISBN 952-5336-66-2 (PDF)
ISSN 1456-176X