TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka

Tutkintotyö

Teemu Moisio

TAMPEREEN AMMATTIKORKEAKOULUN TUNNUSHALLINNAN TIETOKANTA

Työn ohjaaja Ohjelmistotekniikan lehtori Pekka Pöyry
Työn teettäjä TAMK, tietokonekeskus, valvojana DI Petteri Jekunen

TAMPEREEN AMMATTIKORKEAKOULU
Tietotekniikka
Ohjelmistotekniikka
Moisio Teemu

Tampereen ammattikorkeakoulun tunnushallinnan tietokanta
Tutkintotyö 34 sivua + 2 liitesivua
Työn ohjaaja Ohjelmistotekniikan lehtori Pekka Pöyry
Työn teettäjä TAMK, tietokonekeskus, valvojana DI Petteri Jekunen
Huhtikuu 2005
Hakusanat käyttäjähallinto, tietokanta, tietokantaliittymä, käyttäjätietokanta

TIIVISTELMÄ

Tampereen ammattikorkeakoulun tietokonekeskus ylläpitää noin 8000 aktiivista käyttäjätunnusta, joilla on mahdollista päästä käyttämään tietokonekeskuksen tarjoamia palveluita. Käyttäjätunnusten määrän ollessa näinkin iso on mahdollisimman hyvin pyritty automatisoimaan käyttäjätunnusten hallinnan koko elinkaari. Elinkaari tässä tapauksessa tarkoittaa vaiheita tunnuksen luonnista sen poistamiseen lopullisesti järjestelmästä. Automatisoidulla käyttäjätunnusten hallinnalla saadaan poistettua huomattavan paljon itseään toistavaa käsityötä. Käyttäjätunnusten hallinnassa käytetyiltä ohjelmilta vaaditaan tarkkuutta ja varmatoimisuutta. Ylläpitoon liittyvien tietovarastojen tietojen pitää olla luotettavia ja paikkansapitäviä, jotta tunnusten hallinnassa käytetyt ohjelmat voivat toimia oikein.

Tässä työssä on mahdollisimman hyvin pyritty kehittämään käyttäjätunnusten hallintaan tietovarasto, jossa voidaan säilyttää automatisoitujen prosessien ja käyttäjätunnusten hallinnan tarvitsemia tietoja. Toisena työn päätarkoituksena oli toteuttaa tietovarastolle käyttöliittymä, jolla käytetyt tunnusten hallintaohjelmat pääsevät muokkaamaan tietovaraston tietoja. Työn yhtenä sivutuotteena muodostui myös erilaisia Unix-komentorivityökaluja suunnitellun tietovaraston käyttäjien kannalta ystävälliseen ja helppoon hallintaan.

Työssä on esitelty Tampereen ammattikorkeakoulun järjestelmiä käyttäjätunnusten hallinnan näkökulmasta ja selvitetty, minkälaiseen ympäristöön suunnitellut tietokanta ja tietokantaliittymä toteutettiin. Lisäksi tässä työssä on käyty läpi keskeisimmät toteutustavat ja ominaisuudet toteutetuista tietokannasta ja tietokantaliittymästä. Työn lopussa on vielä kerrottu tietokannan käyttöönotosta.

TAMPERE POLYTECHNIC
Computer Engineering
Software Engineering
Moisio Teemu
Tampere Polytechnics Account Management Database
Engineering Thesis 34 pages, 2 appendices
Thesis Supervisor Pekka Pöyry (Lecturer in Software Engineering)
Commissioning Company Tampere Polytechnic, Computer Center, Supervisor: Petteri Jekunen (MSc)

April 2005
Keywords account management, account database, database program

ABSTRACT

The Computer Center of Tampere Polytechnic maintains approximately 8000 user accounts. When the number of the user accounts is so big, management of the accounts has been implemented with automated programs. The programs used in management need to be very reliable and stable. The programs used in the account management needs also very reliable databanks.

In this study, the main purpose was to evolve the account management. This task consists of developing an account management database and an interface to control the database. In addition, this work generated few UNIX tools for maintaining the database.

First, this thesis will describe the environment where the account management database and programs where designed. After that there will be some main issues about the developed database and its development. At the end of this document the database interface and the commissioning of the account management database will be explained

TAMPEREEN AMMATTIKORKEAKOULU TUTKINTOTYÖ
Tietotekniikka
Ohjelmistotekniikka
Teemu Moisio

ALKUSANAT

Tämä työ on tehty kesän 2004 ja kevään 2005 välisenä aikana Tampereen ammattikorkeakoulun tietokonekeskukselle.

Kiitokset erityisesti yhteistyöstä tietokonekeskuksen erikoissuunnittelijalle Petteri Jekuselle, joka hyvällä ammattitaidollaan luotsasi tätä työtä kohti lopputulosta. Kiitokset haluan esittää myös Riku Itäpurolle ja Jarmo Sorvarille laajasta Linux- ja Unix-tuntemuksesta.

Tampereella 11. huhtikuuta 2005

Teemu Moisio

SISÄLLYSLUETTELO

1 JOHDANTO

Tutkintotyön tarkoituksena oli suunnitella ja toteuttaa tunnushallinnan tietokanta relaatiotietokantana sekä Perl-tietokantaliittymä tietokantaan. Tutkintotyö tehtiin Tampereen ammattikorkeakoulun tietokonekeskuksen käyttöön. Tutkintotyön päätarkoitus tietokonekeskukselle oli käyttäjätunnusten hallinnan automatisoinnin kehitys.

Käyttäjätunnusten hallinta Tampereen ammattikorkeakoulun tietokonekeskuksen järjestelmissä on hyvin pitkälle automatisoitua tunnuksen luonnista sen poistamiseen järjestelmästä. Automaatio on toteutettu erilaisilla käyttäjätunnusten hallintaohjelmilla. Käyttäjätunnusten hallinnan automaatiolla vältytään liialliselta käsityöltä esimerkiksi käyttäjätunnuksia poistettaessa järjestelmästä. Käyttäjien hallinnassa automaatioastetta ei varsinaisesti pyritty nostamaan tunnushallinnan tietokannan kehittämisellä, mutta sen käyttöönotto tuo mukanaan sellaisia ominaisuuksia, joilla on mahdollista nostaa automaatioastetta tietyissä prosesseissa. Tunnushallinnan tietokanta ylläpitää siis tunnuksen hallintaan liittyvien prosessien tietoja ja itse tunnuksen.

Työssä tutustutaan ensin yleiseen käyttäjähallintoon, käyttäjätunnusten hallintaan ammattikorkeakouluympäristössä ja lähtökohtana olleeseen järjestelmään. Tietokannan suunnittelun vaiheisiin, toteutukseen ja ongelmakohtiin sekä tietokannan tarjoamiin kehitysmahdollisuuksiin paneudutaan myös. Loppuosassa käydään läpi tietokantaliittymän pääosat ja suunnittelun näkökohtia. Aivan lopussa paneudutaan vielä tietokannan käyttöönottoon.

Tässä kirjallisessa esityksessä selitetään tehdyn työn keskeisimmät osa-alueet, mutta ei syvällisesti paneuduta niiden varsinaiseen toteutukseen.

1.1 Kehitysympäristö

Tietokannan kehitysympäristöksi valittiin MySQL-tietokannan versio 4.0.20, joka asennettiin aluksi Debian-käyttöjärjestelmään testaustietokannaksi. Linux-käyttöjärjestelmä valittiin, koska MySQL-tietokannan lopullinen sijoituspaikka oli Sun Solaris 9 -käyttöjärjestelmällä toimiva palvelinkone. Tärkein valintakriteeri

MySQL-version 4.0.20 valintaan oli se, että se tukee automaattisesti InnoDB-tauluja /2/.

Tietokantaliittymän toteutukseen Perl-kielellä päädyttiin, koska lähes kaikki tietokonekeskuksen käytössä olevat hallintaohjelmat on toteutettu Perl-ohjelmointikielellä. Ohjelmointiympäristönä toimi koko työn tekemisen ajan koulun keskeinen Unix-palvelin, jossa muutkin tunnushallinnan ohjelmat toimivat.

1.2 Työn eteneminen prosessina

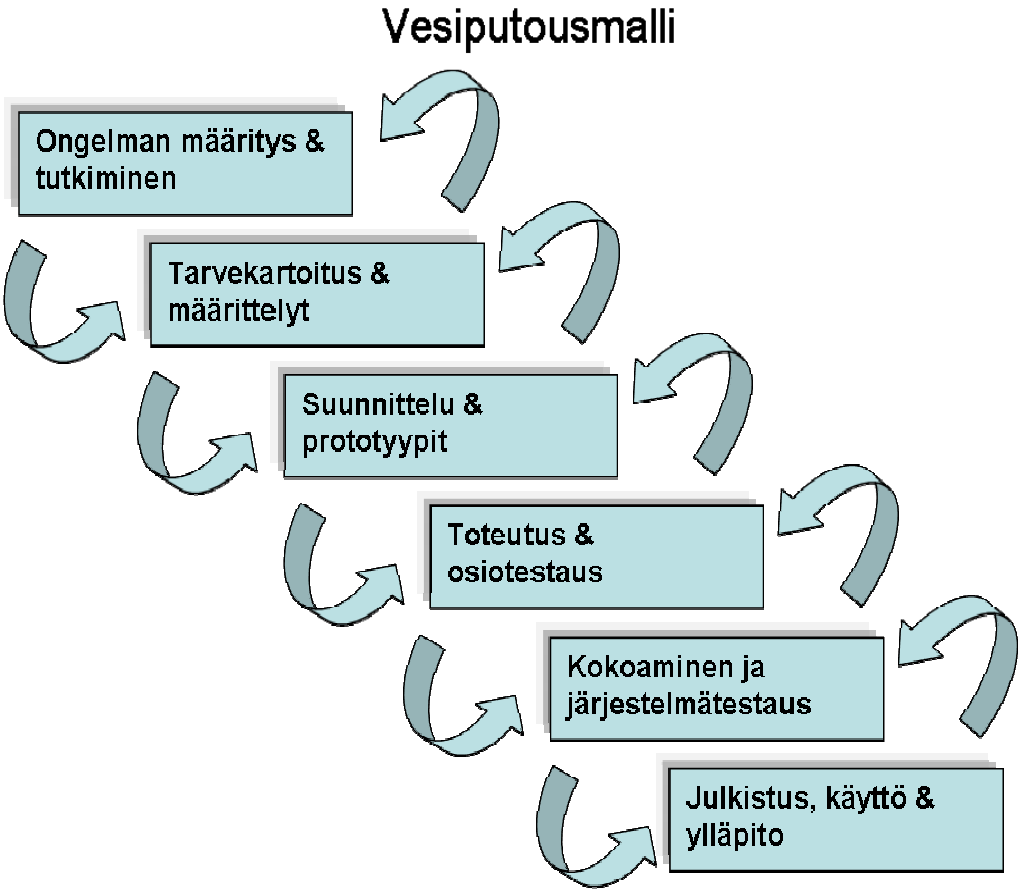
Työ eteni kahdessa päävaiheessa. Ensimmäisessä vaiheessa toteutettiin tietokanta ja toisessa tietokantaliittymä. Tietokannan ja tietokantaliittymän toteutukselle oli varattu aikaa neljä kuukautta. Suunniteltu aikataulu saavutettiin hyvin.

Tietokannan ja tietokantarajapinnan toteuttaminen omina prosesseinaan eteni noudattaen ohjelmistokehityksessä käytettyä vesiputousmallia. Vesiputousmalli etenee kuvan 1 ylimmäisestä laatikosta alemmille tasoille järjestyksessä /1/. Vesiputousmallissa voidaan osavaiheiden välillä palata myös taaksepäin, ja todellisessa tilanteessa vesiputousmallissa liikutaan myös vaiheista toisiin mielivaltaisesti, mikä ei tietenkään ole suotavaa. Vesiputousmalli kuvaa yksinkertaistettuna ohjelmiston elinkaarta.

Vesiputousmalli sopi hyvin toteutettavaan tietokantaan ja tietokantaliittymään, koska molemmat muodostivat oman suoraviivaisesti etenevän prosessin. Molemmissa työ alkoi perehtymisestä järjestelmään sekä suunnittelusta ja päättyi ylläpitoon. Ylläpito ei kuulunut varsinaisesti tehtyyn tutkintotyöhön, mutta muodosti mielenkiintoisen osa-alueen.

Ohjelmistokehityksen eteneminen vaiheittain ja huolella tehty suunnittelu parantavat huomattavasti lopputuloksen laatua. Alustavaan suunnitteluun käytetty aika maksaa itsensä takaisin työn edetessä toteutukseen. Tietokannan suunnitteluun käytettiin hyvin paljon aikaa tässä työssä, mikä käyttöönottovaiheessa ja tietokantaliittymän suunnitteluvaiheessa vähensi suurien muutoksien tarvetta tietokantaan. Tietokannan alustavista rakenteista suunnitteluvaiheessa muodostettiin monia erilaisia ER-kaavioita, joiden kautta mietittiin tietokannan sopivuutta haluttuihin ominaisuuksiin.

Työn toteuttaminen yhden henkilön tekemänä projektina antaa tekijälle itselleen entistä enemmän vapauksia toteutustavassa, kun vielä jaetaan projekti ajallisesti eri osioihin eri osatoteutuksien kannalta ja noudatetaan itselle asetettuja aikarajoja. Niin saadaan määrätietoisesti projekti etenemään hyvin.



Kuva 1 Ohjelmistokehityksessä yleisesti käytetty vesiputousmalli /1/

2 KÄYTTÄJÄHALLINNON PROBLEMATIIKKAA

Käyttäjähallinto tarkoittaa niitä toimenpiteitä ja mekanismeja, joilla organisaatio pitää kirjaa tietojärjestelmiensä käyttäjistä ja heidän käyttöoikeuksistaan /8/. Käyttäjähallinnossa henkilön varmentaminen ja oikeus tarjottuihin palveluihin ovat hyvin tärkeällä sijalla. Suunniteltaessa uusia järjestelmiä, joissa tarvitaan yksilöityjä käyttäjätunnuksia eri käyttäjille, pohditaan varmasti, minkälaisia palveluita käyttäjille on tarkoitus tuottaa. Yksinkertaisimmillaan tämäntyyppinen järjestelmä voisi olla yksi työasema, johon on luotu omat käyttäjät. Monimutkaisimmillaan voidaan ajatella

esimerkiksi monien koulujen palveluiden yhdistämistä, jolloin käyttäjällä olisi mahdollisuus siirtyä henkilöllisyyden varmentamisen jälkeen käyttämään toisen koulun palveluita. Yksinkertaisemmassa esimerkissä käyttäjille tarjotaan työasemalle asennettujen ohjelmien palveluita. Käyttäjiä hallitsee yksinkertaisessa tapauksessa muutama henkilö, jotka vaikuttavat esimerkiksi käyttäjätunnuksien voimassaoloon ja oikeuksiin. Laajemmissa järjestelmissä, kuten korkeakouluympäristössä käyttäjille voidaan tarjota monenlaisia palveluita ja yleensä käyttäjiä hallitsevat useat henkilöt. Opintosihteerit päivittävät esimerkiksi sellaisia tietoja, joista selviää, tarvitseeko käyttäjä vielä palveluihin oikeuttavaa käyttäjätunnustaan. Järjestelmän ylläpitäjät hallitsevat käyttäjätunnuksia, esimerkiksi päivittämällä niiden tietoja tai poistamalla niitä järjestelmästä.

Yksinkertaisessa tapauksessa, kun käyttäjien käyttämä järjestelmä pohjautuu yhteen järjestelmään voi ongelmaksi muodostua riippuvuus käyttöjärjestelmästä ja huono muokattavuus omiin tarpeisiin. Suurissakin organisaatioissa tällainen yhden työaseman esimerkki voidaan nähdä, kun suurehkossa järjestelmässä käytetään vain yhden valmistajan tuotteita palveluiden tuottamiseen ja käyttäjien hallintaan. Esimerkki tällaisesta voisi olla pelkkä Windows-tuotteiden käyttö. Toinen etenemistapa isoissa organisaatioissa on käyttää monien valmistajien tuotteita sekaisin. Erilaisten tuotteiden ja palveluiden käyttö voi pahimmillaan aiheuttaa käyttäjälle sellaisen tilanteen, jossa tarvitaan eri käyttäjätunnukset eri palveluihin. Ratkaisuna erilaisten järjestelmien käyttöön yhdessä on esimerkiksi samojen käyttäjätunnuksien luominen eri järjestelmiin ja automaattisesti toteutettu salasanojen synkronointi. Käyttämällä erityyppisiä järjestelmiä käyttäjähallinnossa voidaan yleensä valita markkinoilta omiin tarkoituksiin sopivia tuotteita ja palveluita entistä paremmin. Järjestelmä mutkistuu yleensä, kun yhdistetään erilaisten valmistajien tuotteita. Käyttämällä eri valmistajien käyttäjähallinnon tietovarastoja saadaan kuitenkin järjestelmästä entistä helpommin muokattava ja laajennettava.

3 KÄYTTÄJÄHALLINTO TAMPEREEN AMMATTIKORKEAKOULUSSA

Tampereen ammattikorkeakoulun tietokonekeskuksen hallinnoimissa järjestelmissä on noin 8000 aktiivista käyttäjätunnusta. Tunnukset ovat LDAP-hakemistossa ja Windows-aktiivihakemistossa (Active Directory). Eri hakemistoihin on siis luotu

sama käyttäjätunnus. Tunnushallinnan tietokannan käyttöönotossa kaikki tunnukset siirrettiin myös tietokantaan. Tietokanta muodosti siis niin sanotun kopion LDAP-hakemiston sisällöstä relaatiotietokantana. Tietokantaan ei kuitenkaan kopioitu sellaista tietoa, joka ei ollut olennaista käyttäjätunnuksien hallinnan kannalta.

Poistettuja käyttäjätunnuksia, joita säilytetään vielä jonkin aikaa tunnuksen poistamisen jälkeen, oli järjestelmässä noin 6000. Tunnushallinnan tietokannan käyttöönoton jälkeen tietokanta ylläpiti noin 15000 käyttäjätunnusta.

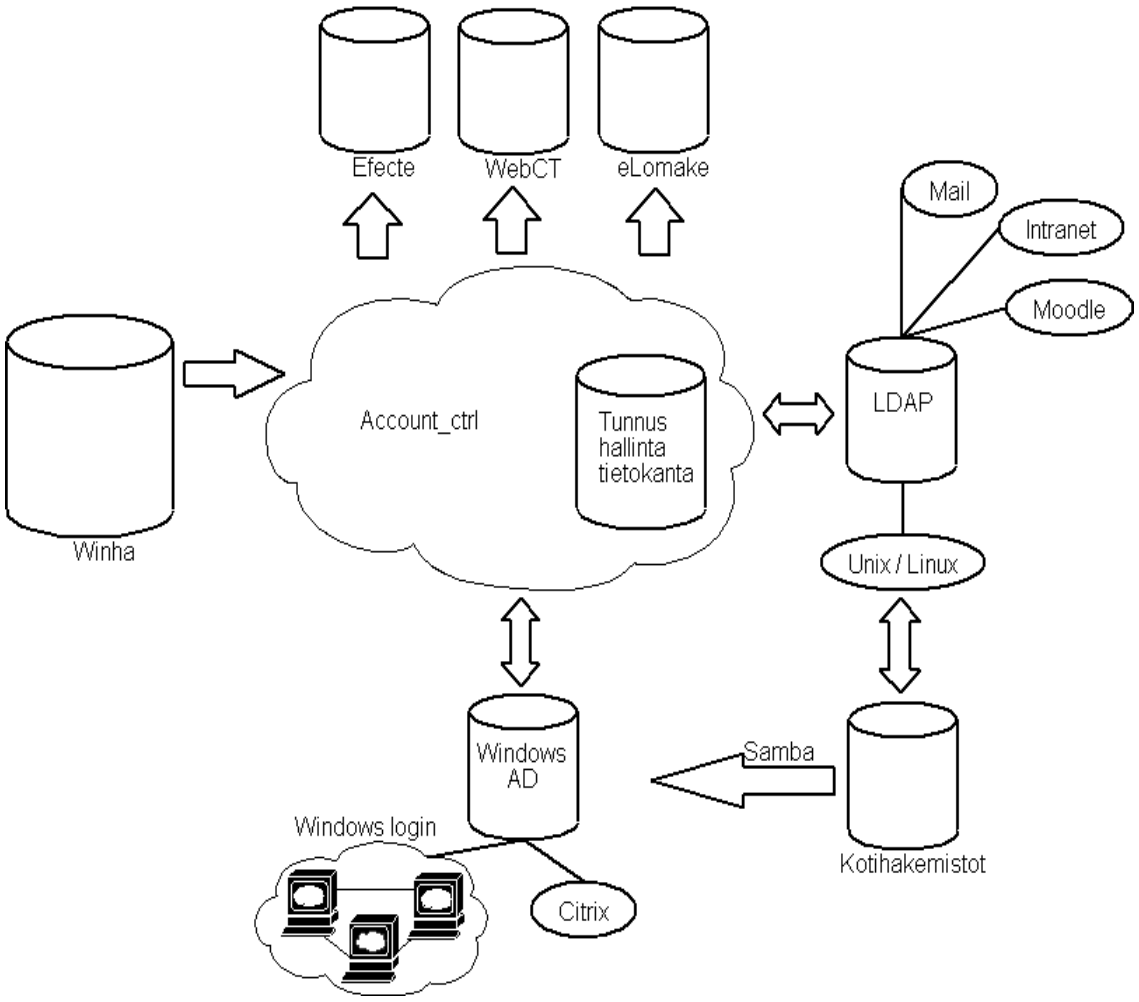
3.1 Ympäristö

Tietokonekeskus tarjoaa palveluita opiskelijoille, opettajille ja muulle henkilökunnalle. Tarjottavia palveluita ovat esimerkiksi sähköposti, koulun oma intranet, ylläpitoon tarkoitettu Efecte, sähköisiin lomakkeisiin käytetty eLomake, etäkäyttöön Citrix ja oppimisympäristöistä mainittakoon Moodle. Tarjottuja palveluita lisätään, parannetaan ja poistetaan käyttäjien ja ympäristön tarpeiden mukaisesti. Järjestelmä on siis hyvin muuttuva lukuun ottamatta keskeisiä elimiä. Tietokonekeskuksen toiminnan on koko ajan yritettävä parantaa järjestelmiään ja tarjoamiaan palveluita vastaamaan sen hetkistä kysyntää ja tarvetta.

Tietokonekeskuksen periaatteena on, että yksi ja sama käyttäjätunnus toimii kaikissa tarjottavissa palveluissa. Yhden käyttäjätunnuksen periaate helpottaa varsinaisia käyttäjiä, koska heidän ei tarvitse muistaa monia käyttäjätunnuksia ja salasanoja käyttäessään palveluita /3/. Yhden käyttäjätunnuksen periaatteeseen on pyritty myös monissa muissa yhteisöissä, joissa on käytössä monia eri palveluita käyttäjille. Tampereen ammattikorkeakoulussa yhden käyttäjätunnuksen periaate on toteutettu luomalla sama tunnus sekä Windows-aktiivihakemistoon että LDAP-hakemistoon. Muuttuvia käyttäjien salasanoja synkronoidaan Windows-aktiivihakemiston ja LDAP-hakemiston välillä ajettavalla ohjelmalla.

Käyttäjätunnuksien hallinnassa käytetyt ohjelmat ovat vuorovaikutuksessa tarjottaviin palveluihin ja tietolähteisiin kuvan 2 osoittamalla tavalla. Tunnushallinnan tietokanta on piirretty kuvaan osaksi keskeistä tunnusten hallintaa. Se siis tarjoaa oman tiedon tallennus- ja säilytyspaikan tunnushallinnan ohjelmille.

Winha-rekisteri kertoo yleensä tavallisille tunnuksille niiden voimassaoloajan, siksi se on piirretty kuvaan 2 hyvin isoksi ja merkittäväksi osaksi järjestelmää. LDAP-hakemistopalvelin toimii tiedonlähteenä Unix-koneille, sähköpostille, intralle eli koulun sisäiselle verkkosivustolle ja oppimisympäristö Moodlelle. Käyttäjien kotihakemistot sijaitsevat suurimmaksi osaksi Unix-palvelimilla, mistä ne tuodaan Windows-puolelle Samba-jaoksi.



Kuva 2 Työn tekohetkellä ollut ammattikorkeakoulun ympäristö. Kuva on kuvattuna tunnushallinnan näkökulmasta. Nuolet osoittavat tietovirran suunnan.

3.2 Mahdolliset käyttäjätunnustyypit

Laajassa käyttöympäristössä, joka tarjoaa paljon erilaisia palveluita, on tarpeellista luoda käyttäjätunnuksia niiden käyttötarkoituksen mukaisesti. Luomalla tunnuksia niiden tyypin mukaisesti rajoitetaan usein tietyillä tunnustyypeillä tunnuksen käytössä

olevia resursseja. Tunnuksien erilaiset tyypit voidaan Tampereen ammattikorkeakoulun ympäristössä yleensä jakaa neljään eri luokkaan. Tavallisten opiskelijoiden tunnukset ovat yksi ryhmä ja ne muodostavat tunnuksista suurimman osan. Opettajat ovat seuraavaksi isoin käyttäjätunnusten ryhmä. Muut käytössä olevat tunnukset voidaan jaotella affiliate- ja extranet-tyyppeihin englanninkielisillä termeillä. Affiliate-tunnus tehdään sellaiselle henkilölle, joka on osa ammattikorkeakoulun toimintaa, mutta ei varsinaisesti kuulu päätoimintaan eli esimerkiksi siivoojan tai vahtimestarin käyttäjätunnus. Extranet- tunnus on käytössä sellaisella henkilöllä, jonka pitää päästä koulun tarjoamiin palveluihin käsiksi, mutta henkilö ei varsinaisesti ole sidoksissa koulun toimintaan. Extranet- ja affiliate-tunnuksille yleensä luodaan tunnus rajoitetummilla resursseilla kuin esimerkiksi normaalia henkilökuntatunnusta luotaessa. Esimerkiksi käyttäjätunnuksen resursseja voidaan rajoittaa poistamalla tunnukselta kokonaan oikeus käyttää Windows-työasemia.

3.3 Käyttäjätunnuksen voimassaoloajan ilmeneminen järjestelmässä

Tunnuksien tilasta kertova tieto, kuten voimassaoloaika on hajautettuna muutamaan paikkaan Tampereen ammattikorkeakoulun järjestelmissä. Tunnuksen voimassaolo voi olla kerrottuna Winha-rekisterissä tai mahdollisesti paikassa, jossa voidaan tunnukselle määritellä poikkeuksellinen voimassaoloaika. Tavallisesti kuitenkin Winha-rekisteri on se lähde, jonne on tallennettu tieto, kuinka pitkään tunnus on voimassa. Tunnuksien poikkeuksellinen voimassaoloaika oli määriteltynä tiedostoon ennen tunnushallinnan tietokannan toteutusta.

Ensimmäisiä asioita, joiden toteutuksesta tunnushallinnan tietokantaan keskusteltiin, olivat juuri nämä poikkeuksellisen voimassaoloajan omaavat tunnukset.

3.4 Tunnuksen elinkaari

Käyttäjätunnuksen elinkaari alkaa siitä vaiheesta, kun se ensimmäisen kerran luodaan järjestelmään. Ennen tunnuksen luomista henkilön tiedot on jo normaalissa tapauksessa tallennettu Winha-rekisteriin ja käyttäjä on saanut omat käyttäjätunnuksensa sinne. Kun Winha-rekisteriin on tallennettu tarvittavat tiedot

uudesta henkilöstä, voidaan uusi käyttäjätunnus luoda tietokonekeskuksen tarjoamille palveluille.

Käyttäjätunnuksella on käytössään kaikki ne resurssit, joita tunnuksella on määritelty koko sen ajan kun tunnuksen voimassaoloaika on kunnossa Winha-rekisterissä. Tunnuksen voimassaoloajan umpeuduttua Winha-rekisterissä aloitetaan tietokonekeskuksen tarjoaman tunnuksen poistaminen järjestelmästä. Käyttäjätunnus poistetaan käytöstä kymmenen päivän kuluttua siitä, kun käyttäjän opiskeluoikeus on umpeutunut Winha-rekisterissä. Tunnuksen tietoja säilytetään pisimmillään noin kaksi vuotta sen poistamisesta ja sen jälkeen tunnus häviää kokonaan järjestelmästä. Tunnuksen elinkaarikin päättyy siihen, kun tunnus on kokonaan poistettu järjestelmästä.

3.5 Tunnukseen sitoutuvat resurssit

Normaalilla käyttäjätunnuksella on käytössään kaikki mahdolliset resurssit, kuten käyttäjän oma kotikansio, sähköposti, pääsy koulun omille intranet-sivuille, jaettujen kansioiden käyttömahdollisuus, Moodle-oppimisympäristö, Unix-palvelut, Windows-työasemien käyttömahdollisuus ja Citrix-etätyömahdollisuus. Ainoastaan, kun luodaan sellaisia tunnuksia, joiden käyttötarkoitus poikkeaa normaalista, rajoitetaan tunnukseen sitoutuvia resursseja.

3.6 LDAP-hakemistopalvelin

LDAP-hakemistopalvelimella sijaitsevat normaalisti kaikki koulun käyttäjätunnukset. Hakemistohierarkia tukee hyvin ammattikorkeakouluympäristöä, jolloin erilaisia osakokonaisuuksia voidaan sijoittaa omiin hakemistohaaroihinsa. Käyttäjätunnukset sijaitsevat LDAP-hakemistorakenteessa omissa hakemistoissaan, joissa ne on jaoteltu omiin lohkoihinsa opiskelijoiden, henkilökunnan ja muiden käyttäjätunnuksiin.

Tunnushallinnan tietokannasta muodostui sen käyttöönoton jälkeen LDAP-hakemiston kopio tunnuksien osalta. Tunnushallinnan tietokantaan ei viety kaikkea mahdollista tietoa LDAP-palvelimelta ja tietokannassa kaikki tunnukset sijaitsevat

loogisesti samassa lohkossa. Tunnushallinnan tietokannassa tunnukseen liittyviä tietoja on pilkottu eri tauluihin niiden asiayhteyteen kuuluvalla tavalla.

3.7 Automatisoitu poistoprosessi

Käyttäjätunnukselle aloitetaan poistoprosessi voimassaoloajan päätyttyä joko Winha-rekisterissä tai poikkeuksellisen voimassaolon määrittelypaikassa. Poistoprosessin ohjelma ajetaan päivittäin kaikille tunnuksille. Poistoprosessi on toteutettu tilakoneen avulla, jossa prosessissa oleva käyttäjätunnus liikkuu tilojen välillä. Eri tiloissa suoritetaan erilaisia poistoon liittyviä toimenpiteitä.

Aloitettaessa tunnuksen poistoprosessia käyttäjälle lähetetään sähköposti ilmoituksena prosessin aloittamisesta. Poistoprosessin ensimmäisissä vaiheissa tunnuksen sisältämien resurssien paikkatietoja kerätään, koska seuraavassa vaiheessa, kun kymmenen päivää on kulunut poistamisen aloittamisesta, kerättyjen paikkatietojen avulla tunnus poistetaan käytöstä ja tietojen varmuuskopiointi aloitetaan.

3.8 Poistoprosessin toteutus ennen tunnushallinnan tietokantaa

Ennen tunnushallinnan tietokannan toteutusta poistoprosessi käytti tietolähteenään tiedostoja. Tiedoston nimi sisälsi käyttäjätunnuksen, poistoprosessin tilan, mahdolliset alatilat ja poistopäivämäärät. Tiedoston sisään oli tallennettuna tunnukseen liittyvien resurssien tietoja, kuten esimerkiksi sähköpostiosoite.

Esimerkki kuvitellusta poistoprosessitiedostosta olisi seuraavan näköinen.

t0tmoisi-BACKUP-0-2-20040612-20040703.txt

Tiedostot oli luokiteltu kahteen eri kansioon, joista toisessa oli opiskelijoiden ja toisessa henkilökuntaan kuuluvien poistoprosessi. Poistoprosessiohjelma ajettiin opiskelijoille ja henkilökunnalle eri vaiheissa. Poistoprosessia ajettiin myös eri tavalla jo poistetuille tunnuksille ja vasta poistoprosessiin otettaville tunnuksille.

Tunnushallinnan tietokanta korvasi käyttöönoton jälkeen tiedostot tietolähteenä ja kehityksen aikana poistoprosessin ohjelmat yhtenäistettiin niin, että samaa ohjelmaa

ajettiin samaan aikaan poistoprosessissa oleville ja vielä poistamattomille tunnuksille. Yhtenäistäminen ei varsinaisesti liittynyt tunnushallinnan tietokantaan, mutta selvensi huomattavasti poistoprosessia.

3.9 Poikkeustunnuksien toteutus ennen tietokantaa

Tietokonekeskuksen järjestelmissä on tarvetta ylläpitää sellaista tietolähdettä, jonne voidaan määritellä halutuille käyttäjätunnuksille poikkeuksellinen voimassaoloaika. Ennen tunnushallinnan tietokannan toteutusta tämän tietolähteen muodosti tiedosto, jonka sisälle oli määritelty tällaiset tunnukset omille riveilleen. Yksi kuvitteellinen tiedoston rivi voisi näyttää seuraavalta.

t0tmoisi:students::20040922: Teemun työtunnus

Rivi alkoi aina käyttäjätunnuksella, jota seurasi erotinmerkki eli kaksoispiste tässä tapauksessa. Seuraavaksi tunnuksesta kerrottiin, onko tunnus opiskelija- tai muuntyyppinen tunnus. Esimerkissä oleva päivämäärä kertoo, mihin asti tunnus on voimassa. Viimeinen kenttä oli kommentointia varten. Tiedostossa oli käytössä myös omana kommentointimerkkinä #, jolla voitiin saada tiedoston rakenteesta entistä selvempi kommentoimalla tiettyjä tunnusryhmiä. Tiedosto käytiin aina läpi, kun tarkistettiin, olisiko tarkasteltavalle tunnukselle merkitty poikkeuksellinen voimassaoloaika tiedostoon.

3.10 Perl-ohjelmointikielen käyttö järjestelmän hallinnassa

Perl-ohjelmointikieli tarjoaa valmiita hyviä moduuleja esimerkiksi MySQL-tietokannan kanssa käytettäväksi samantyyppisesti kuin PHP-ohjelmointikieli. Tekemäni tietokantaliittymä toteutettiin käyttämällä Perl-moduulia, DBI:mysql. Moduuli tarjoaa paljon palveluita MySQL-tietokantaan. Palveluita käyttämällä on kohtuullisen helppoa muodostaa omia funktioita ja moduuleita, jotka sopivat hyvin omaan haluttuun käyttötarkoitukseen.

Ohjelmointikielenä järjestelmän hallinnassa Perl sopii hyvin tiedostojen ja prosessien hallintaan. Perl-kielen tehokkuus ilmenee hyvin myös tekstiä parsittaessa. Lyhyillä ohjelmilla voidaan saada paljon aikaan verrattuna esimerkiksi C-kieleen.

Järjestelmän hallinnassa Perl-kielen parhaat puolet ilmenevät kielen komentorakenteen takia. Ohjelmakoodista tulee helposti ymmärrettävää ja havainnollista. Perl-kieli on myös helposti omaksuttavissa jonkin toisen kielen rinnalle.

4 TUNNUSHALLINNAN TIETOKANTA

4.1 Suunnittelun lähtökohdat

Ensimmäisissä tietokannan rakenteen suunnittelupalavereissa järjestelmästä muodostui kolme hyvin vahvaa käsitettä, joita olivat henkilö, henkilön rooli koulussa ja henkilölle kuuluva käyttäjätunnus. Muodostuneet käsitteet pysyivät samoina työn alusta sen loppuun asti, saaden työn edetessä hieman erilaisia merkityksiä. Käsitteistä muodostui silti hyvin suoraviivainen tapa lähteä toteuttamaan tietokannan rakennetta. Tietokannan suunnittelua vaikeutti eniten se, että järjestelmässä on hyvin paljon poikkeuksia. Tunnuksia voidaan luoda ja poistaa erittäin mielivaltaisesti, mikä on kuitenkin tarpeellista tietyissä tilanteissa. Toisena esimerkkinä poikkeuksista kaikista koulun henkilöistä ei ollut saatavilla tarpeeksi yksilöiviä henkilötietoja, jotka olisivat auttaneet saamaan tietokannasta yksiselitteisemmän. Tietokannan yksiselitteisyyttä olisi huomattavasti parantanut esimerkiksi se, jos jokaisesta tietokantaan tallennettavasta henkilöstä olisi ollut saatavilla henkilötunnus. Kaikista henkilökuntaan kuuluvista tai ulkomaalaisista opiskelijoista ei ollut saatavilla henkilötunnuksia.

Tietokannan rakenteesta yritettiin toteuttaa mahdollisimman yleiskäyttöinen, mutta silti mahdollisimman paljon järjestelmää mallintava. Rakenteen suunnittelussa otettiin huomioon jatkossa suunniteltuja muutoksia ja kehitysratkaisuja, kuten toisen tunnusavaruuden tuominen tietokantaan. Tietokantaan tallennettiin aluksi tietokonekeskuksen ylläpitämät tunnukset omaksi tunnusavaruudeksi. Toisen tunnusavaruuden tuominen tietokantaan tarkoittaisi esimerkiksi uuden käyttäjätunnus-ryhmän lisäämistä tietokantaan. Uudella käyttäjätunnusryhmällä tarkoitetaan tässä tilanteessa esimerkiksi sellaisia tunnuksia, joiden voimassaoloa hallitaan eri paikassa kuin muita tietokonekeskuksen tunnuksia. Myös uusi käyttäjätunnusryhmä voisi olla täysin erillään Tampereen ammattikorkeakoulun toiminnoista, muodostaen näin oman

kokonaisuutensa. Erillään olevien tunnuksien ylläpito voitaisiin silti hyvin toteuttaa samassa tietokannassa. Tietokannassa eri tunnusavaruudet sijaitsevat samassa tietokantataulussa eriytettynä toisistaan omilla määritelmillään.

Tietokonekeskuksen palveluita käyttävälle henkilölle voi sitoutua yksi tai useampi käyttäjätunnus järjestelmään, koska henkilö voi toimia esimerkiksi opettajan ja opiskelijan roolissa. Molempiin rooleihin voi siis sitoutua eri käyttäjätunnus, jos esimerkiksi tarvitaan erilaisia oikeuksia henkilökuntatunnuksille tai opiskelijatunnukset tarvitaan kokeilu- tai testauskäyttöön.

Järjestelmässä on myös mahdollista, että yksi käyttäjätunnus kuuluu useammalle henkilölle eli tunnus on esimerkiksi ylläpitotunnus, jota käytetään opiskelijaoikeuksien saamiseen tietyissä tilanteissa. Yleiskäyttöiset ylläpitotunnukset eivät ole hyvä ratkaisu, koska esimerkiksi sellaisessa tilanteessa, jossa käyttäjätunnusta käyttänyt henkilö eroaa ja tietää tunnuksen salasanan niin ehdottomasti tietoturvan takia käyttäjätunnukseen pitäisi vaihtaa uusi salasana. Ongelmaksi tällaisissa tilanteissa muodostuu salasanan vaihtamisen unohtaminen tai vaihdetun salasanan tiedottaminen muille ylläpitotehtävistä vastaaville.

4.2 MySQL-tietokannan valinnan perusteet

MySQL-tietokantaa käytetään tietolähteenä monissa verkkosovelluksissa. MySQL sopii hyvin tämän tyyppisiin sovelluksiin ilmaisuutensa ja hyvien kehitysmahdollisuuksiensa vuoksi. Isoissa tuotantokäyttöön tarkoitetuissa sovelluksissa turvaudutaan usein kaupallisiin ja järeämpiin tietokantoihin, joiden odotetaan käsittelevän isoja tietomääriä ja tietokantakyselyjä lyhyissä ajoissa. Tunnushallinnan tietokannan ei varsinaisesti tarvitse olla erittäin nopea, kunhan tietokanta säilyttää tallettamansa tiedon eheänä.

Suunnittelun alkuvaiheissa ei ollut aivan varmaa, kuinka paljon tunnuksia tietokantaan tultaisiin tallentamaan. Tallennettavien tunnuksien määrän ei kuitenkaan uskottu kasvavan niin suureksi, ettei MySQL-tietokanta pystyisi niitä ylläpitämään. Käyttöönottovaiheessa tunnushallinnan tietokanta ylläpiti noin 15000 käyttäjätunnuksen tietoja. Käyttäjätunnuksien määrä ei varmastikaan muodostu

pullonkaulaksi vaan mahdollisesti se, jos tietokantaa ruvetaan käyttämään tulevaisuudessa laajemmin tietolähteenä, ja päällekkäisten kyselyjen määrä kasvaa.

MySQL-tietokannasta saatavilla oleva dokumentaatio on hyvin kattavaa ja valmiita esimerkkejä on paljon. Tietokantaan oli myös helppo muodostaa yhteydet Perl-ohjelmointikielestä. Esimerkkejä löytyy kattavasti myös siitä, kuinka tietokantaa voidaan mahdollisimman paljon hyödyntää käytetyllä ohjelmointikielellä. Valmiita funktioita tietokannan muokkaamiseen ja kyselyiden tekemiseen on myös paljon tarjolla Perl-ohjelmointikielessä.

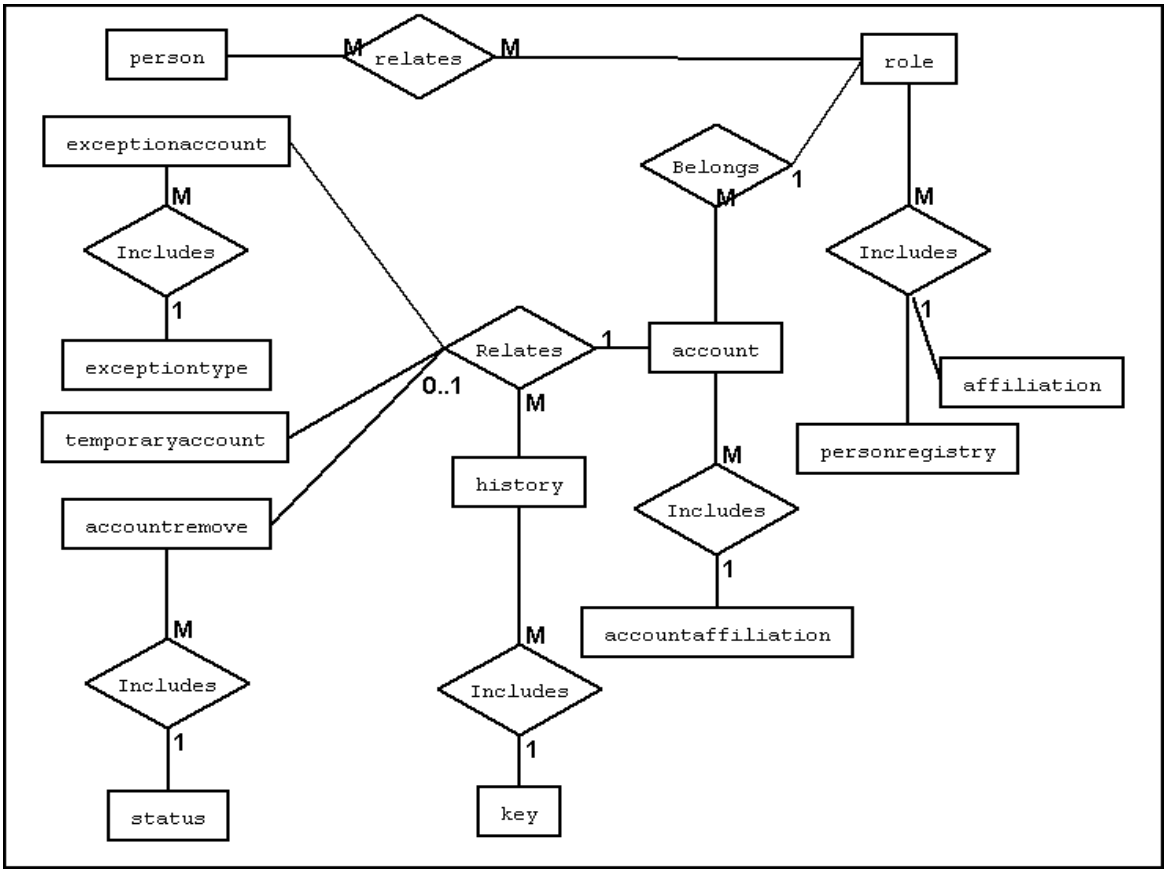
4.3 Tietokannan rakenne ja ER-malli

ER-mallilla kuvataan tietokannan taulujen suhteita toisiinsa /9/. Mallista näkyy myös taulujen attribuutit. Relaatiotietokannoissa taulujen suhteiden kuvaamisella ja dokumentoimisella on hyvin tärkeä osa suunnittelussa, toteutuksessa ja ylläpidossa. Valmiista tietokannasta, jossa on esimerkiksi käytetty sellaisia tauluja missä suhteet toisiin tauluihin eivät näy esimerkiksi vierasavaimina, on erittäin vaikea jälkikäteen löytää liittymiä toisiin tauluihin. ER-malli on hyvä lähtökohta tietokannan suunnittelussa. Tämän työn ensimmäinen suunnittelu kuukausi käytettiin suurimmaksi osaksi erilaisten ER-mallien tekemiseen. Valmiiseen ER-malliin on helppo ajatella erilaisia käyttötapauksia ja kuinka ne sopivat ER-mallin tarjoamaan ratkaisuun.

Tunnushallinnan tietokanta englanninkieliseltä lyhenteeltään Adb (Account Database), koostuu osaksi kolmesta tärkeimmästä päätaulusta, joita ovat henkilö (person), rooli (role) ja käyttäjätunnus (account). Päätaulut muodostavat tietovaraston relaatiotietokantana, jota voidaan verrata koulun LDAP-hakemistopalvelimeen. Tietokannassa ylläpidetään ainoastaan huomattavasti vähemmän tietoa kuin LDAP-hakemistopalvelimessa.

Seuraavaksi tärkeimmät taulut tietokannassa päätaulujen lisäksi ovat tunnus-tauluun liittyvät taulut, tunnuksien poistoprosessi (accountremove) ja poikkeustunnukset (exceptionaccount). Tietokannassa on myös määrittely-tauluja, joihin on vakioitu tiettyjä haluttuja arvoja. Jatkokehityksessä tietokantaan tehtiin käyttäjäkohtaiselle historiatiedolle (history) tarkoitettu taulu ja tilapäistunnuksien (temporaryaccount) hallintaan käytettävä taulu.

Kuva 3 on esittää edellä mainittujen taulujen liittymisen toisiinsa. Relaatiot on toteutettu tietokannassa vierasavaimilla. Tietokantaa pitää käyttää vierasavaimien mukaisesti eli esimerkiksi käyttäjätunnusta ei voi tuoda tietokantaan, jos sitä ei pystytä liittämään johonkin jo tietokannassa olevaan roolin.



Kuva 3 Tietokannan ER-malli. Taulujen attribuutit on poistettu kuvan yksinkertaistamisen vuoksi.

Henkilön suhde rooliin on toteutettu monen suhde moneen tavalla, koska haluttiin että rooliin voidaan liittää monta henkilöä tai ei yhtään ja että myös yhdelle henkilölle voi kuulua monta roolia tai ei yhtään. Rooliin liittyy yksi tai useampi käyttäjätunnus. Käyttäjätunnuksella voi olla poikkeus-, tilapäis-, poistoprosessi- tai historiatietoja omissa tauluissaan. Määrittely-taulut on liitetty toisiin tauluihin yhden suhde moneen tavalla, joten esimerkiksi tunnustauluun voi viedä tunnuksia vain määrittely-taulussa olevalla määreellä.

4.4 Tietokannan taulut

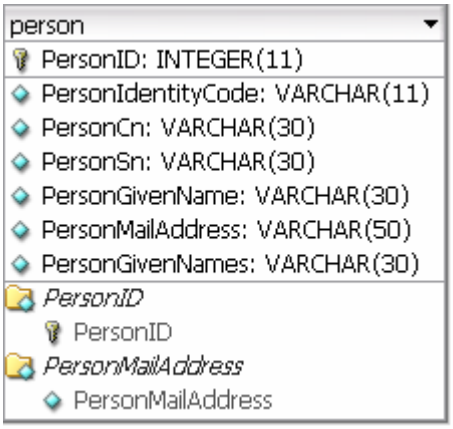
Tietokannan taulut ovat kaikki toteutettu InnoDB-tyyppisinä, koska tietokannassa käytetään vierasavaimia ja hallinnassa transaktioita. Transaktiot mahdollistavat tauluja päivitettäessä mahdollisista virhetilanteista toipumista ja päällekkäisten päivitysten helpomman hallinnan /5/. Tietokannan taulut ja niiden liittyminen toisiinsa on kuvattu liitteessä 1.

InnoDB-tyyppiset taulut ovat huomattavan paljon hitaampia käyttää, kuin MySQL:n MyISAM perustaulut, mutta mahdollistavat tietokannan määrätyn käyttötavan. Suunnittelun alkuvaiheessa tietokanta toteutettiin ensin alustavasti MyISAM-tauluilla, mutta vierasavaimien takia siirryttiin InnoDB-tauluihin. Tietokannan käsittely hidastui huomattavasti InnoDB-tauluihin siirryttäessä.

4.4.1 Henkilö-taulu

Henkilö-taulussa ylläpidetään henkilöille kuuluvia tietoja. Taulussa pääavaimena on käytetty juoksevaa numeroa, joka identifioi henkilöt tietokannassa. Tietokannassa henkilöistä ylläpidetään henkilötunnusta, kutsumanimeä, etunimeä, sukunimeä, kaikkia nimiä ja sähköpostiosoitetta. Ylläpidettäviä henkilötietoja on huomattavan vähän, siihen nähden, mitä kaikkia niitä voisi olla. Halutun käyttötarkoituksen kannalta tässä tietokannassa ylläpidetään mahdollisimman vähän henkilötietoja.

Kuvassa 4 on esiteltynä henkilö-taulu ja mitä tietotyyppejä siinä on käytetty attribuuteille. Kuvasta ilmenee myös indeksoidut attribuutit. Henkilölle kuuluva sähköpostiosoite on indeksoitu tietokantaan tehtävien hakujen nopeuttamiseksi. Tehtäessä hakuja, lisäyksiä tai muutoksia henkilö-tauluun, henkilöt identifioidaan henkilötunnuksen alkuosalla, etunimellä ja sukunimellä. Näin menetellään, koska kaikista koulun henkilöistä ei välttämättä ole saatavilla yksiselitteisiä tietoja muista tietolähteistä tai tunnushallinnan tietokannasta. Testauksella, aikaisemmilla tuloksilla ja johtopäätöksillä ilmeni, että jos henkilötunnuksen alkuosa, etunimi ja sukunimi ovat samat verrattaessa tietokannassa olevaa henkilöä johonkin toisen tietolähteen henkilöön, voidaan olla varmoja että kyseessä on sama henkilö.

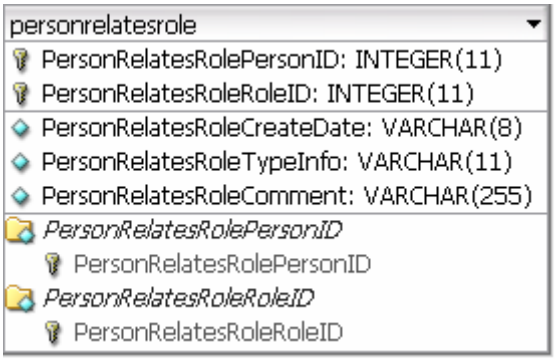


Kuva 4 Henkilö-taulu

4.4.2 Henkilön ja roolin liitostaulu

Monesta moneen suhde henkilöllä ja roolilla on toteutettu tekemällä tietokantaan liitostaulu, jossa ylläpidetään liitettävien taulujen pääavaimia /4/. Suhde toteutettuna erillisellä taululla mahdollistaa sen, että molemmilla liitoksen osapuolilla voi olla rajaton määrä liitoksia toiseen tauluun tai ei yhtään. Samoja suhteita kahden taulun välillä liitostauluun ei voida tuoda, koska pääavaimien käyttö estää sen.

Liitostaululla saadaan myös liitoksesta muodostettua kuvaava liitos, käyttämällä taulussa omia attribuutteja, kuten tässäkin taulussa on tehty. Henkilön ja roolin suhteesta voidaan ylläpitää luontipäivää, suhteen tyyppiä ja kommenttikenttää. Suhteen tyyppiä ja kommenttikenttää ei tietokannassa käytetä hyväksi vielä millään tavalla, mutta ne tehtiin sinne mahdollista käyttöä varten. Kuvasta 5 näkyy myös, kuinka molemmat pääavaimet on indeksoitu. Ylimääräinen liitostaulu aiheuttaa myös lisää ohjelmalogiikkaa tietokantaliittymään.

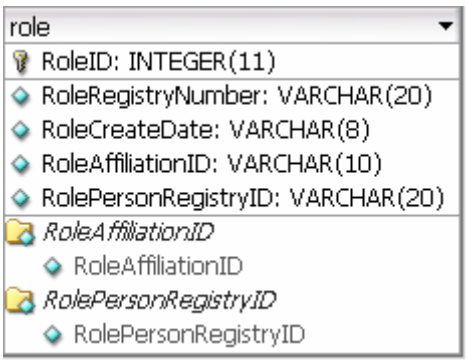


Kuva 5 Liitostaulu, jolla henkilöt on liitetty rooleihin.

4.4.3 Rooli-taulu

Rooli-taulussa ylläpidetään tietoja, jotka kertovat roolin suhteesta (RoleAffiliationID) kouluun ja missä kyseisen roolin voimassaoloaikoja (RolePersonRegistryID) ylläpidetään. Suhteesta ja voimassaoloajan ylläpitopaikasta kertovat tyyppimääreet ovat vierasavaimia määrittely-tauluista. Tauluun ei siis voi tuoda rooleja muilla kuin määrittely-tauluissa määritellyillä avaimilla. Rooli-taulun rekisterinumero (RoleRegistryNumber) identifioi roolia, joka on tavallisilla opiskelijoilla Winha-rekisterin käyttäjätunnus. Rekisterinumero voi myös olla minkä tahansa muun rekisterin käyttäjätunnus tai vain keksitty arvo. Tauluun tallennetaan myös päivämäärä, jolloin rooli on ensimmäisen kerran luotu tietokantaan. Rooli-taulu on esitettynä kuvassa 6. Kuvasta näkyy, kuinka toisesta taulusta otetut vierasavaimet (RolePersonRegistryID, RoleAffiliationID) ovat indeksoitu.

Rooli-tauluun hakuja tehtäessä, haku suoritetaan yleensä rekisterinumerolla, koska se identifioi parhaiten roolia. Hakuja voidaan tietysti toteuttaa roolin identifioivalla numerolla, jos se on jo esimerkiksi haettu tietokannasta muusta taulusta.



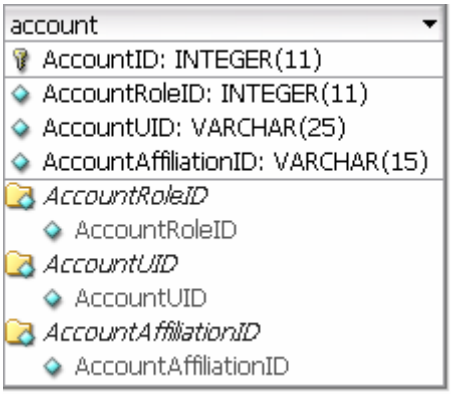
Kuva 6 Rooli-taulu, jolla yksilöidään henkilöiden ja tunnuksien liittymistä koulun toimintaan.

4.4.4 Käyttäjätunnus-taulu

Käyttäjätunnus-taulussa ylläpidetään varsinaista tietokonekeskuksen palveluihin oikeuttavaa käyttäjätunnusta (AccountUID), kuten kuvassa 7 on esitetty. Käyttäjätunnukset identifioidaan tietokannassa juoksevalla numerolla, kuten muissakin päätauluissa on tehty. Taulussa on myös yhtenä kenttänä tunnuksen liittyminen koulun toimintaa (AccountAffiliationID), joka on vierasavain määrittely-taulusta. Rooli-taulun ja käyttäjätunnus-taulun yhdestä moneen suhde on toteutettu

ottamalla rooli-taulusta roolin identifioiva numero vierasavaimeksi käyttäjätunnus-tauluun.

Käyttäjätunnus-taulussa ylläpidettävällä liityntä (AccountAffiliationID) määritteellä mahdollistetaan se, että samaan rooliin voidaan liittää esimerkiksi opiskelija- ja henkilökuntatunnus, mikä oli toivottu ominaisuus tietokantaa suunniteltaessa.



Kuva 7 Käyttäjätunnus-taulu

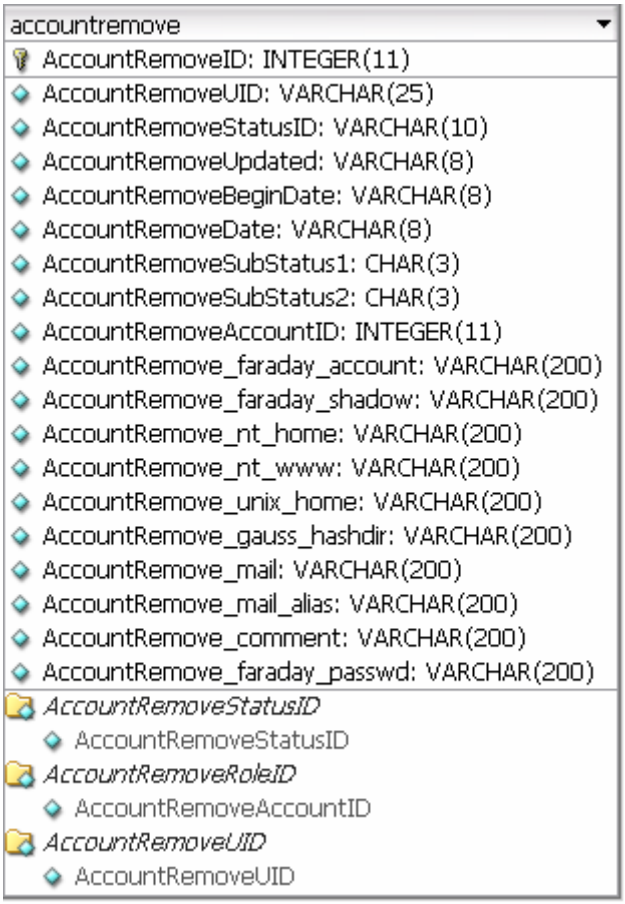
Käyttäjätunnus-taulussa eri tunnusavaruudet erotetaan myös omikseen liityntä (AccountAffiliationID) määritteellä. Tampereen ammattikorkeakoulun opiskelija esimerkiksi luokiteltaisiin TPU_STUDENT tunnisteella. Tunnisteessa etuliite kuvaa tunnusavaruutta ja loppuosa opiskelijana esiintymistä.

4.4.5 Poistoprosessi-taulu

Poistoprosessi-taulussa ylläpidetään poistoprosessiin otettuja käyttäjätunnuksia. Prosessissa olevat tunnukset identifioidaan juoksevalla numerolla (AccounRemoveID), joka on taulun pääavain. Varsinaiseen käyttäjätunnus-tauluun tämä taulu on liitetty ottamalla käyttäjätunnus-taulusta (account) vierasavain (AccountRemoveAccountID) tähän tauluun, mikä on tunnus-taulussa pääavaimena.

Poistoprosessissa olevasta tunnuksesta ylläpidetään sen statusta (AccountRemoveStatusID), joka on vierasavain yhdestä määrittely-taulusta. Taulussa ylläpidetään myös päivityspäivämäärää (AccountRemoveUpdated), poiston aloittamispäivämäärää (AccountRemoveBeginDate), varsinaista poistopäivämäärää (AccountRemoveDate) ja poistoon liittyviä ala-statuksia (AccountRemoveSubStatus1,

AccountRemoveSubStatus2). Muut attribuutit, joiden nimissä esiintyy alaviivoja, kuten kuvassa 8 on esitetty, liittyvät tunnukseen sitoutuviin resursseihin.

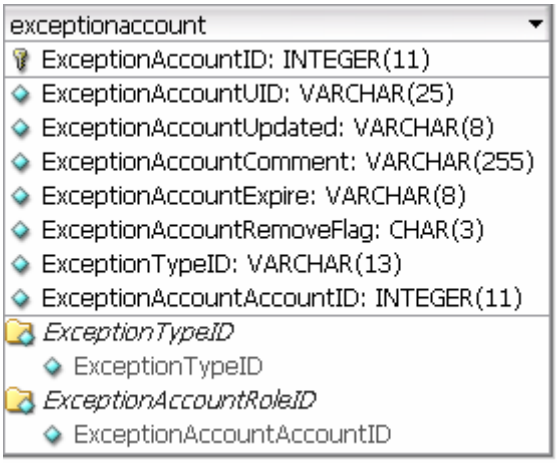


Kuva 8 Poistoprosessi-taulun rakenne

4.4.6 Poikkeustunnus-taulu

Poikkeustunnukset identifioidaan myös juoksevalla numerolla, kuten on tehty kaikissa muissakin tietokannan päätauluissa. Tässä taulussa ylläpidetään poikkeustunnuksen päivityspäivämäärää, kommentointia, voimassaolostatusta (ExceptionAccountExpire), tyyppiä, varsinaisen käyttäjätunnuksen identifioivaa numeroa ja poistoon vaikuttavaa erikoiskenttää (ExceptionAccountRemoveFlag). Erikoiskenttä ei tullut vielä käyttöön tässä työssä. Kuvassa 9 on esitetty taulun attribuutit ja niiden tyypit. Liitos tunnus-tauluun on tehty ottamalla tunnus-taulun pääavain vierasavaimeksi tähän tauluun. Poikkeustunnuksien tyypin määrittely-taulusta on myös otettu vierasavain tähän tauluun. Poikkeustunnus-tauluun ei voi tuoda tunnuksia muilla kuin poikkeustunnuksien tyypin määrittely-taulussa olevilla määritteillä.

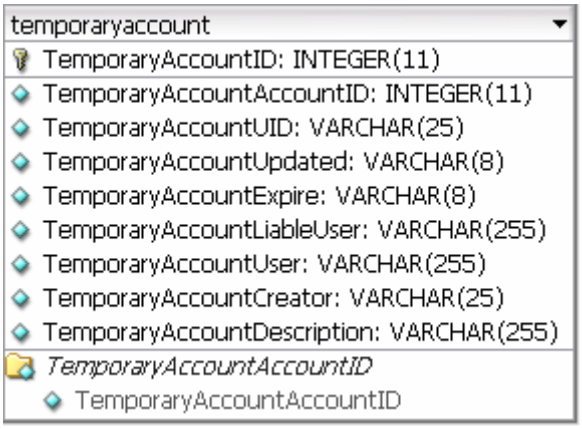
Poikkeuksellisia voimassaoloaikoja tarkastellaan, kun ajetaan kaikille tunnuksille tunnushallinnan poistoprosessiohjelmaa. Poikkeustunnuksien hallintaan ja muokkaamiseen kehitettiin käyttäjäystävällinen komentorivityökalu helpottamaan ylläpitotyötä.



Kuva 9 Poikkeutunnus-taulu

4.4.7 Tilapäistunnus-taulu

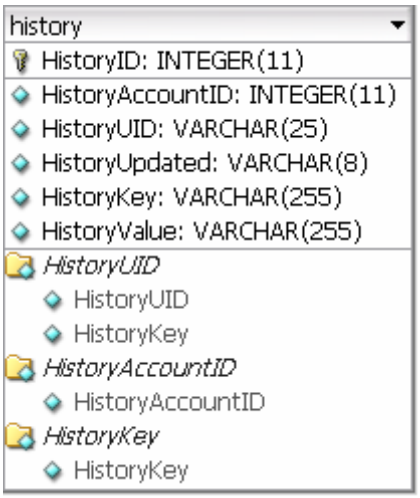
Tilapäistunnus-taulu on niille tunnuksille, jotka kuuluvat tilapäistunnusten ryhmään. Tilapäistunnuksia käyttävät sellaiset henkilöt, jotka tarvitsevat tilapäisiä tunnuksia lyhyeksi ajaksi. Tilapäistunnus-taulu liittyy varsinaiseen tunnukseen taulussa olevalla tunnus-taulusta otetulla vierasavaimella (TemporaryAccountAccountID), kuten kuvassa 10 on esitetty. Tilapäistunnuksille on oma taulu, koska niitä hallitaan omassa prosessissaan. Taulussa ylläpidetään tilapäistunnuksista identifioivaa juoksevaa numeroa (TemporaryAccountID), päivityspäivämäärää, viimeistä voimassaolopäivää (TemporaryAccountExpire), vastuullista henkilöä (TemporaryAccountLiableUser), varsinaista käyttäjää (TemporaryAccountUser), tunnuksen tekijää (TemporaryAccountCreator) ja sitä käyttötarkoitusta (TemporaryAccountDescription), mihin tunnus on tehty.



Kuva 10 Tilapäistunnus-taulu

4.4.8 Historiatieto-taulu

Käyttäjäkohtaista historiatietoa ylläpidetään tässä taulussa. Taulussa on vierasavaimena yhdestä määrittely-taulusta (key) sen pääavain. Määrittely-taulussa on kerrottuna, minkälaisia historiatietoja tässä taulussa voidaan ylläpitää. Historia-taulun pääavaimena toimii yhdistetty avain käyttäjätunnuksesta ja avainarvosta (HistoryKey) eli yhdelle käyttäjätunnukselle ei voi viedä kahta kertaa samalla avainarvolla tietoja. Arvo (HistoryValue) kertoo avaimen arvon. Historia-taulua käytettiin Unix-ohjelmassa, jotta saataisiin käyttäjälle edelliset käytetyt arvot esiarvoiksi valinnoille. Kuvassa 11 on esitetty historia-taulun attribuuttien tyypit.



Kuva 11 Historia-taulu

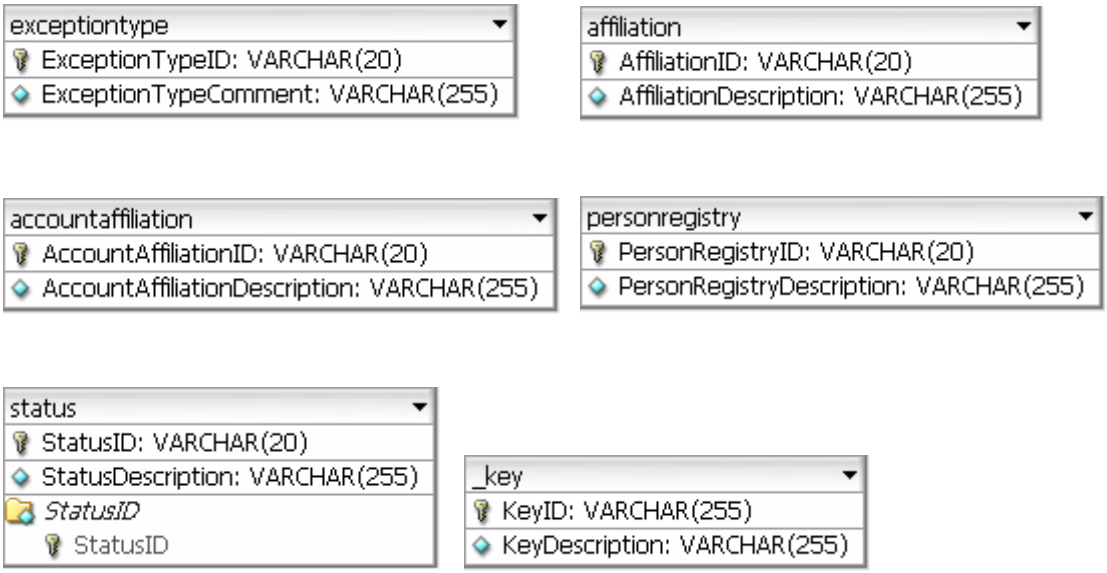
4.4.9 Määrittely-taulut

Poikkeustyyppien (exceptionType) määrittely-taulussa on määriteltynä mahdollisia poikkeustyyppejä poikkeustunnuksille. Määrittely-taulu poikkeustyypeille tehtiin, koska aiemmin käytetyssä tiedostossa poikkeustunnukset oli kategorisoitu omiin luokkiinsa. Poikkeustyyppi-taulussa ylläpidetään poikkeustunnuksen tyyppiä taulun pääavaimessa, joka on vierasavain poikkeustunnus-tauluun. Kommentointi-kenttä (ExceptionTypeComment) on tarkoitettu poikkeustyypin selitykseen. Kuvassa 12 näkyy, että jokaisen määrittely-taulun tyyppikenttä on merkkijono, millä saadaan aikaan tyypistä jo itsestään kuvaava määre. Kaikissa tauluissa on myös kommentointi-kenttä. Poikkeustyyppien-tauluun on tallennettuna mahdollisiksi tyypeiksi esimerkiksi FROM_LUO ja CC_SYSADM. CC_SYSADM tyyppiin kuuluvat tunnukset ovat esimerkiksi ylläpitotehtävissä käytettyjä tunnuksia, joita käyttää tietokonekeskuksen henkilökunta.

Henkilön liittymisestä koulun toimintaan on olemassa kaksi eri määrittely-taulua. Ensimmäinen taulu (affiliation) on liitetty rooli-tauluun. Tässä taulussa mahdollisia tyyppejä ovat esimerkiksi STAFF tai STUDENT. Toinen (accountaffiliation) taulu on liitetty käyttäjätunnus-tauluun. Tämän taulun tyypit ovat muuten samanlaisia, mutta etuliitteellä kerrotaan mahdollinen tunnusavaruus esimerkiksi TPU_STUDENT.

Toinen rooli-tauluun liitetty taulu (personregistry) kertoo, missä roolin voimassaoloaikaa hallitaan. Tässä taulussa määriteltyjä tyyppejä ovat esimerkiksi TAMK_WINHA ja TAO_WINHA. Poistoprosessin-tauluun liitetyssä määrittely-taulussa (status) on määriteltynä poistoprosessin tilakoneen tiloja esimerkiksi ALERT_N ja BACKUP.

Historia-tauluun liitetyssä määrittely-taulussa (key) on määriteltynä mahdollisia avainarvoja, joita voidaan tallentaa historia-tauluun. Taulussa on esimerkiksi pääavaimina description ja liable_user.



Kuva 12 Kaikki tietokannassa käytetyt määrittely-taulut

4.5 Tietokannan tarjoamat kehitysratkaisut

Suunnittelun lähtökohtana oli suunnitella tietokanta, jossa voidaan ylläpitää käyttäjätunnuksien poistoprosessia ja järjestelmässä olevia poikkeustunnuksia. Ensimmäinen jälkikehityksessä toteutettu kehitysratkaisu oli tilapäistunnusten hallinnan automatisointi. Tilapäistunnuksille ajetaan päivittäin hallintaohjelmaa, joka käyttää tietolähteenään tunnushallinnan tietokantaa ja poistaa tilapäistunnuksen käytöstä, jos sen voimassaoloaika on umpeutunut tietokannassa. Käytännössä ohjelma vaihtaa käyttäjätunnuksen salasanan ja saa näin aikaan tunnuksen poistumisen käytöstä silloiselta käyttäjältä.

Tilapäistunnuksien käyttöön tehtiin käyttäjäystävällinen ohjelma, jolla ylläpitäjän on helppo ottaa tilapäistunnus käyttöön tarvittaessa. Tilapäistunnukset voidaan luoda ohjelmalla käyttöön enimmillään kymmeneksi päiväksi ja tilapäistunnukselle on aina pakotetusti nimettävä vastuuhenkilö, mikä vaadittiin ohjelman toiminnallisuudelta /6/.

Käyttäjien historiatietojen ylläpito oli myös toinen jatkokehitysratkaisu, joka toteutettiin jälkikehityksessä. Tilapäistunnusten luonnissa käytetty ohjelma tallettaa tunnushallinnan tietokantaan luotavan tilapäistunnuksen tiedot ja ohjelmaa käyttävän henkilön täyttämät tiedot henkilölle omiksi historiatiedoikseen. Seuraavan kerran tilapäistunnus ohjelmaa käytettäessä käyttäjällä on esiasetuksina aikaisempia käyttämiään tietoja.

Tunnushallinnan tietokantaan voidaan hyvin sijoittaa sellaisia kehitysratkaisuja, joiden ei haluta sitoutuvan tärkeimpiin ja keskeisimpiin tietovarastoihin. Tunnushallinnan tietokanta muodostaa oikein käytettynä tietovaraston kaikista Tampereen ammattikorkeakoulun käyttäjätunnuksista. Käyttäjätunnuksiin voidaan helposti sitoa erilaisia prosesseja, kuten esimerkiksi poistoprosessi on liitetty käyttäjätunnuksiin.

4.6 Tietokannan käytön tarjoamat parannukset

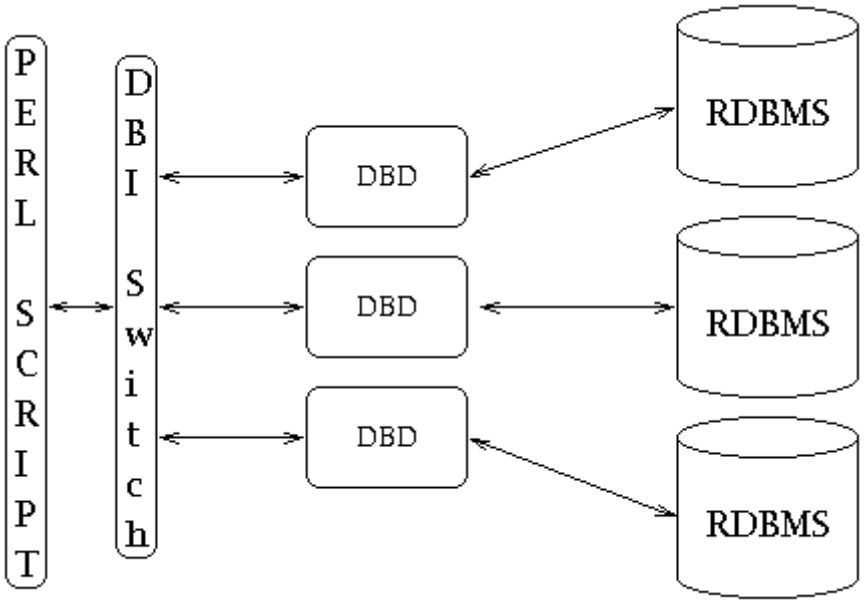
Yhtenä tietokannan käytön tuomana parannuksena voidaan mainita nopeampi tietojenkäsittely tietokannassa kuin tiedostojärjestelmässä. Tietokannan nopeus voi kärsiä tulevaisuudessa, jos tietokantaa ruvetaan käyttämään laajemmin tiedonlähteenä muille sovelluksille.

Oma näkökulmani tietokannan tarjoamiin parannuksiin on, että tiedostojärjestelmän käyttö tietokantana ei ole hyvä ratkaisu. Varsinkin kun on kyse monista tuhansista pienistä tiedostoista, joita käytetään tiedonlähteenä. Tietokanta tarjoaa hyviä palveluita tiedon järjestelyyn ja erilaisten hakujen toteuttamiseen. Myös raporttien teko hyvin toimivasta tietokannasta on helpompaa kuin esimerkiksi aikaisemmin käytetyistä tiedostoista. Yhteen relaatiotietokantaan saadaan myös helposti koottua relaatioiden avulla kasattuja erilaisia kokonaisuuksia.

5 PERL-TIETOKANTALIITTYMÄ

5.1 Rajapinnan asettamat vaatimukset

Tietokantaliittymä tpumysql-moduuli tarjoaa tunnushallinnan tietokannan hallintaan suunniteltuja palveluita. Rajapinta vaatii toimiakseen Unix- tai Linux-ympäristön, jossa on Perl-kääntäjä asennettuna. Tietokantaliittymä käyttää muista Perl-kielen valmiista moduuleista DBI-moduulia (DataBaseInterface), joka on yleisesti käytetty tietokantaliittymä /7/. DBI-moduuli tarjoaa oman moduulinsa DBD::mysql (DataBaseDriver) MySQL-tietokantaan. Kuvassa 13 on kuvattu DBI-moduulin toimintahierarkiaa. Kuvasta näkyy, kuinka DBI-moduuli tarjoaa rajapintoja moniin eri tietokantoihin.



Kuva 13 Tietokantaliittymässä tarvitun DBI-moduulin rakennekuva /7/

DBD::mysql moduuli tarjoaa palveluita yhteyden muodostamiseen, purkamiseen ja tietokannan muokkaamiseen. Moduulissa on myös paljon muitakin hyödyllisiä palveluita. Yksi erittäin hyödylliseksi muodostunut palvelu oli funktio *quote*. *Quote* funktiolla pystyy muokkaamaa merkkijonoja sellaisiksi, että MySQL ei tunnista ohjausmerkkejä kuten esimerkiksi ”\n”. Windows-puolen kotihakemistoissa kenoviivana käytetään ”\”, jonka MySQL tunnistaa ohjausmerkin aloittavaksi merkiksi.

Tietokantaliittymä käyttää tietokonekeskuksen omista moduuleista tpu_def.pm moduulia, jonne on käytettäviä vakioita määriteltynä.

5.2 Rajapinnan sijoittuminen järjestelmään

Tietokantaliittymää käytetään tietokonekeskuksen omissa moduuleissa winha.pm, acc_ctrl.pm ja backup_acc.pm. Winha-moduulin palveluita käytetään käyttäjätunnuksien käyttöoikeuksia tarkistettaessa. Tunnushallinnan pääohjelmien muut palvelut sijaitsevat acc_ctrl.pm ja backup_acc.pm moduuleissa. Tietokantaliittymää käytetään myös tunnusten luonti- ja poistotyökaluissa. Pääsääntönä rajapintaa liitettäessä ensin olleeseen järjestelmään oli korvata muista

moduuleista kaikki sellaiset kohdat, joissa kirjoitettiin tai luettiin tiedostojärjestelmästä.

5.3 Rajapinnan kuvaus

Tietokantaliittymä tarjoaa palveluita yhteydenhallintaan ja taulujen tietojen päivittämiseen. Julkisilla tietojen asetus-funktioilla voi viedä uusia tietoja ja myös päivittää vanhoja. Tietokantaliittymän rakenne on kuvattu paremmin tietokantaliittymän kuvauksessa liite 2.

Käyttäjän ei tarvitse huolehtia yhteyden muodostamisesta käyttäessään tietokantaliittymän palveluita. Yhteys muodostetaan aina käytettäessä julkisia palveluita, jos sitä ei ole aikaisemmin avattu. Yhteyden sulkemisesta käyttäjän pitäisi huolehtia. Yhteyden sulkeminen ei ole kuitenkaan aivan pakollista käyttäjän toimesta, koska käyttämättömät yhteydet poistuvat automaattisesti järjestelmästä, jos niitä ei käytetä.

Rajapinnan julkiset funktiot on kaikki nimetty etuliitteellä tpumysql, mikä helpottaa tpumysql-moduuliin kuuluvien funktioiden löytämistä muista Perl-ohjelmista.

6 TUNNUSHALLINNAN TIETOKANNAN KÄYTTÖÖNOTTO

6.1 Korvattavien prosessien ja muiden tietojen siirtäminen tietokantaan

Tietokannan käyttöönottovaiheessa siirrettiin tietokantaan kaikki tunnukset LDAP-hakemistopalvelimelta. Poistoprosessin ja poikkeuksellisen voimassaoloajan omaavat tunnukset siirrettiin myös tietokantaan. Tietojen siirto tapahtui Perl-ohjelmilla. Ohjelmat käyttivät tietokannan muokkaamiseen tietokantaliittymän tarjoamia palveluita.

Kaikkien tunnuksien haku *populateldap.pl* ohjelmalla tapahtui muodostamalla yhteys hakemistopalvelimeen ja luomalla hakemiston sisällöstä tietorakenteita, jotka sisälsivät kaiken kopioitavan tiedon. Tietorakenteet käytiin seuraavaksi läpi, yksitellen tunnuksien osalta, samalla kopioimalla tunnuksia tietokantaan.

Poistoprosessin ja poikkeustunnuksien vieminen tietokantaan tapahtui *populate.pl* ohjelmalla, joka ensin luki poikkeustunnuksien tiedoston ja poistoprosessien tiedostot ja muodosti niistä omat tietorakenteet, jotka sisälsivät kopioitavat asiat. Tietorakenteet käytiin seuraavaksi läpi kopioimalla samalla tunnuksia tietokantaa.

6.2 Tunnushallinnan tietokannan testaus ja käyttöönoton testaus

Tietokantaliittymälle suoritettiin epävirallista moduulitestausta samalla aikaa, kun se valmistui. Tietokantaliittymä moduulitestattiin vielä epävirallisesti erikseen sen valmistumisen jälkeen. Tietokannan toimivuutta testattiin ennen käyttöönoton testausta kokeilemalla tietojen viemistä tietokantaan ja käyttämällä tietokantaliittymää tietokannan muokkaamiseen.

Varsinainen käyttöönoton testaus tapahtui ajamalla rinnakkain korvattavaa ja uutta tietokantaa tietolähteenään käyttävää järjestelmää. Tämä testausvaihe kesti noin kaksi kuukautta. Testaus perustui tietokannan tietojen ja aikaisemman järjestelmän tietojen vertailemiseen molempien ajojen jälkeen.

7 YHTEENVETO

Yksinkertaisissa järjestelmissä käyttäjähallinnon merkitys on huomattavasti pienempi kuin laajoissa ja monien käyttäjien käyttäjätietoja ylläpitävissä järjestelmissä. Sellaisissa ympäristöissä, joissa käyttäjien vaihtuvuutta on paljon, on tarvetta automatisoida käyttäjähallintoa. Korkeakouluympäristössä on hyvin paljon tarvetta automatisoinnille, koska vuosittain opiskelijoiden vaihtuvuus on hyvin suurta. Uusia opiskelijoita tulee lisää ja vanhoja valmistuu pois koulusta jaksoittain. Lyhytkestoisia kursseja suorittavat opiskelijat tarvitsevat myös omat käyttäjätunnukset. Automatisoitu käyttäjähallinto tarvitsee varmatoimisia ja luotettavia tietolähteitä toimiakseen luotettavasti. Tietolähteet voidaan yhdellä tavalla jakaa karkeasti kahteen ryhmään. Ensimmäinen ryhmä käsittää sellaiset tietolähteet, jotka tukevat varsinaisia korkeakoulun pääprosesseja. Toiseen ryhmään voidaan sijoittaa sellaisia tietolähteitä, jotka sisältävät ylläpidossa tarvittavaa metatietoa. Molemmat jaotellut tietolähteet voidaan nähdä tärkeänä riippuen järjestelmän toiminnallisuudesta ja rakenteesta.

Hyvälle käyttäjähallinnolle tunnusmerkkeinä voidaan nähdä hyvin keskitetyt käyttäjien tietoja ylläpitävät tietovarastot. Erittäin tärkeä seikka hyvässä käyttäjähallinnossa on myös hyvä tietoturva, koska on erittäin tärkeää että käyttäjien henkilökohtaiset tiedot eivät leviä vääriin käsiin. Tässä työssä mahdollisia tietoturvariskejä ei pohdittu tarkasti. Mahdollisessa jatkokehityksessä tulisi huomioida riittävän hyvän tietoturvan varma toteutuminen. Mielestäni tämän työn puitteissa on kuitenkin saatu toteutettua vaadittu toiminnallisuus tietokannan ja tietokantaliittymän osalta.

LÄHTEET

Painetut lähteet

1 Ian Sommerville, Software Engineering Seventh Edition. Addison-Wesley 2004. 66 s.

5 Elmasri & Navathe, Fundamentals of Database Systems Third Edition. Addison-Wesley 2000. 629 s.

9 Elmasri & Navathe, Fundamentals of Database Systems Third Edition. Addison-Wesley 2000. 45 s.

Sähköiset lähteet

2 MySQL AB [www-sivu] [viitattu 14.11.2004] Changes in release 4.0.x (Production) Saatavissa: http://dev.mysql.com/doc/mysql/en/News-4.0.x.html

4 MySQL AB [www-sivu] [viitattu 23.1.2005] An Introduction to Database Normalization Saatavissa: http://dev.mysql.com/tech-resources/articles/intro-to-normalization.html

7 Perl [www-sivu] [viitattu 12.2.2005] About DBI Saatavissa: http://dbi.perl.org/about/

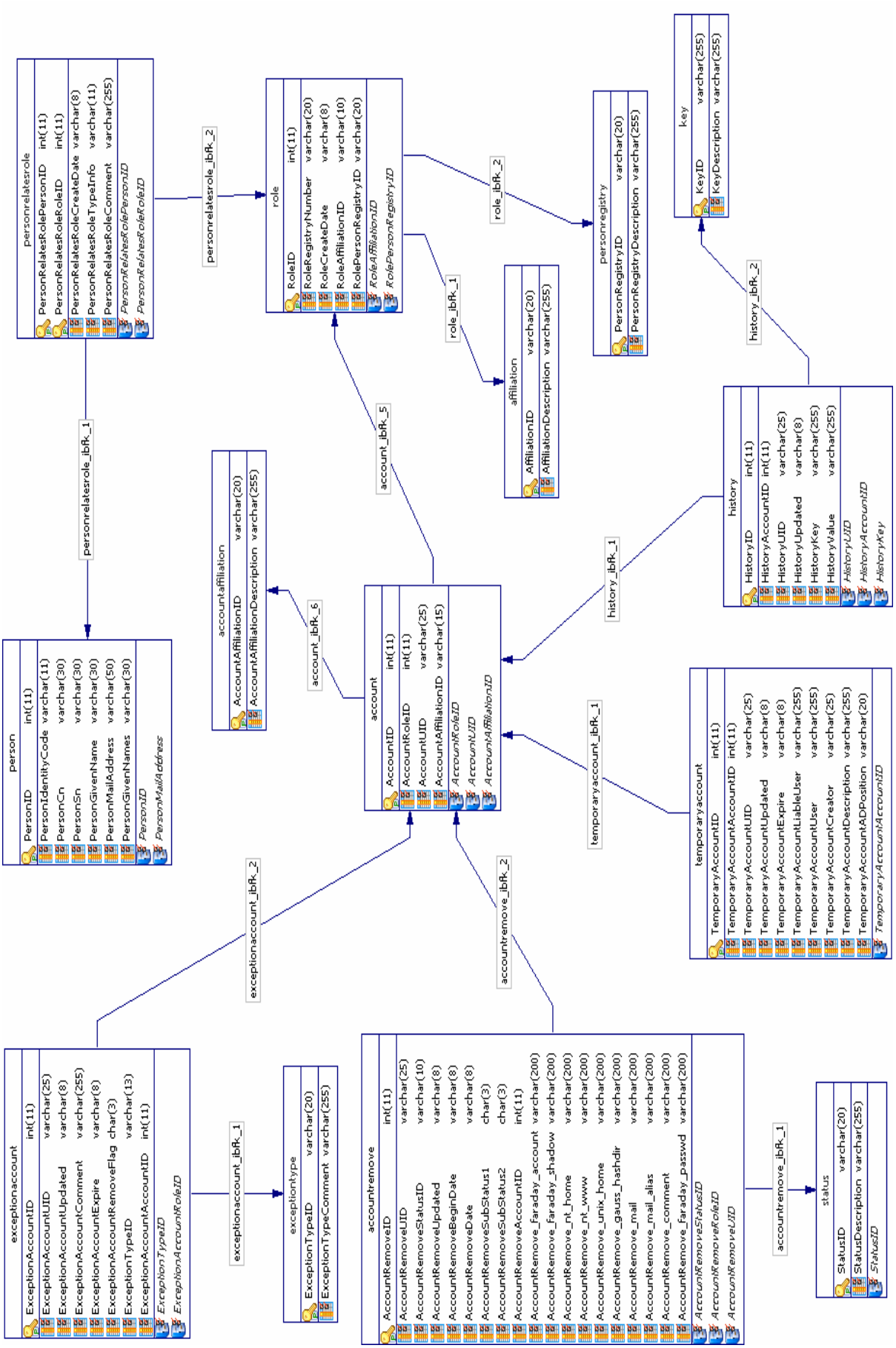
8 Tieteen tietotekniikan keskus [www-sivu] [viitattu 20.2.2005] Käyttäjähallinto korkeakouluissa Saatavissa: http://www.csc.fi/suomi/funet/middleware/index.phtml

Painamattomat lähteet

3 Petteri Jekunen, erikoissuunnittelija, Haastattelu 15.11.2004
Tampereen ammattikorkeakoulu tietokonekeskus

6 Jarmo Sorvari, ATK-järjestelmäpäällikkö, Haastattelu 10.11.2004
Tampereen ammattikorkeakoulu tietokonekeskus

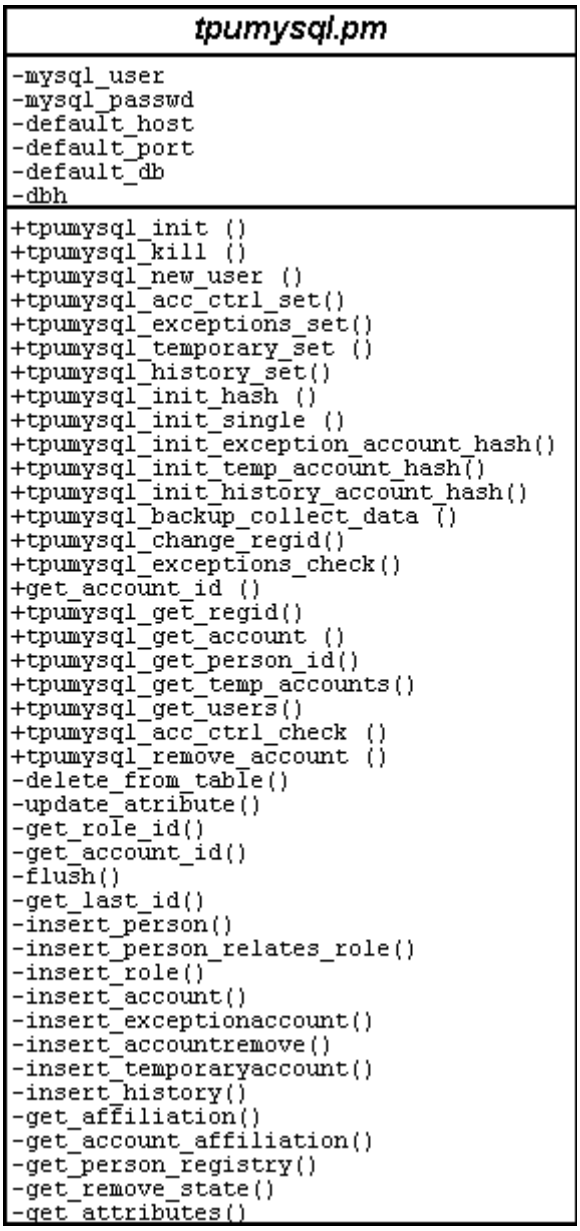
Liite 1 Tunnushallinnan tietokannan rakennekuva



Liite 2 Tunnushallinnan tietokannan tietokantaliittymän kuvaus

Tässä liitteessä on kuvattu esimerkiksi kaikki tietokantaliittymän sisältämät funktiot ja rajapinnan yleiset muuttujat.

Ensimmäisessä laatikossa tietokantaliittymän nimen kentän jälkeen on kuvattuna rajapinnan sisäisiä muuttujia. Miinus-merkki muuttujan tai funktion nimen edessä kuvaa, että muuttuja tai funktio on käytössä vain rajapinnan sisällä. Plus-merkillä erotetut tarkoittavat tietokantaliittymästä muille moduuleille ja ohjelmille näkyviä metodeita tai muuttujia.



Liite 2 Tietokantaliittymä