LAUREA

Materiaalinhallintajärjestelmä uusmediatoimiston tarpeisiin

Case: Barabra Oy

Räsänen, Esko
Sinisalo, Antti

2010 Leppävaara

Laurea-ammattikorkeakoulu
Laurea Leppävaara

Materiaalinhallintajärjestelmä uusmediatoimiston tarpeisiin - Case: Barabra Oy

Esko Räsänen, Antti Sinisalo
Tietojenkäsittelyn koulutusohjelma
Opinnäytetyö
Toukokuu, 2010

Laurea-ammattikorkeakoulu **Tiivistelmä**
Laurea Leppävaara
Tietojenkäsittelyn koulutusohjelma
Tietojärjestelmät

Esko Räsänen, Antti Sinisalo

Materiaalinhallintajärjestelmä uusmediatoimiston tarpeisiin
Tapaus: Barabra Oy

Vuosi 2010 Sivumäärä 73

Tämän opinnäytetyön tavoitteena on toteuttaa yritykselle sopiva www-pohjainen tiedostojenhallintajärjestelmä, jota se voi hyödyntää jokapäiväisessa työelämässä yhdessä sidosryhmien kanssa. Kehitettävän tietojärjestelmän johdosta tiedonsiirto ja -hallinta keskittyy yhteen paikkaan, jolloin myös yrityksen muut toiminnot kuten vanhojen projektien arkistointi helpottuu. Järjestelmän suunnittelutyöhön osallistui opiskelijoiden lisäksi myös kohdeyrityksen työntekijöitä erilaisten projektitapaamisten yhteydessä.

Idea materiaalinhallintajärjestelmän kehittämiseen uusmediatoimiston tarpeisiin alkoi jo työharjoittelujakson yhteydessä vuonna 2008. Työharjoittelussa koettujen haasteiden ja ongelmien pohjalta tarkoituksena on suunnitella ja kehittää juuri kyseiselle yritykselle sopiva tietojärjestelmä palvemaan yrityksen, yhteistyökumppaneiden ja asiakkaiden materiaalinhallintatarpeita.

Opinnäytetyön tarkoituksena on suunnitella ja toteuttaa helppokäyttöinen ja tehokas www-selainpohjainen järjestelmä, jonka avulla kohdeyrityksen työntekijät, yrityksen asiakkaat sekä mahdolliset yhteistyökumppanit voivat lähettää ja vastaanottaa sekä hallita erilaisia projektiresursseja kuten suuria videotiedostoja. Suurimpana haasteena tällaisen järjestelmän kehittämisessä on laaja käyttäjäkunta, jolla on hyvin vaihteleva osaamistaso. Järjestelmän kehitystyön tulee olla hyvin käyttäjälähtöistä.

Olemassa olevia muita järjestelmiä tutkittiin ja tärkeää tietoa saatiin käyttämällä muun muassa benchmarking-menetelmää. Suuri osa olennaisesta tiedosta saatiin yrityksen sisäisistä kokemuksista, joita on kohdattu jokapäiväisessä työssä.

Vertailemalla ja tutkimalla muita järjestelmiä sekä yrityksen tarpeita saatiin aikaan määrittely, jonka mukaisesti suunniteltiin ja ohjelmoitiin toimiva järjestelmä käyttäen erilaisia www-tekniikoita. Järjestelmä otettiin onnistuneesti testikäyttöön vuonna 2010 ja sitä jatkokehitetään edelleen.

Asiasanat: Tietojärjestelmä, materiaalinhallinta, ohjelmointi, benchmarking, käyttöliittymä

Laurea University of Applied Sciences **Abstract**
Laurea Leppävaara
Business information technology programme
Information Systems

Esko Räsänen, Antti Sinisalo

Material management system for a new media agency's needs
Case: Barabra Ltd.

Year 2010 Pages 73

The idea to develop a material management system for a new media agency started during the practical training period in 2008. Based on the challenges and problems experienced during the practical training period, the purpose is to design and develop an information system that is particularly aimed at serving the material management needs of the target company, its partners in co-operation and clients.

The purpose is to create web-based file management software which can be used in everyday work with the reference groups. Due to the developed information system, the transfer and management of files will be centralized, which will also make the target company's other processes easier; for example, archiving the old projects will become more linear.

The thesis is aimed at designing and developing an easy-to-use and efficient system that will run within the users' web-browser, which the target company's workers, clients and possible reference groups can use to send, receive and manage varying types of project resources such as large video files. The greatest challenge in the development process is the varying know-how of the user base. The development will have to be very user-centric.

Other existing file management systems were researched and important information was collected, for example by using the benchmarking technique. A great deal of the gathered information was received from experiences that had been faced during the every day job within the target company.

By researching and comparing other systems as well as researching the company's own needs a specification for the system was created, which was used to design and program fully functional software using various types of web-techniques. The system was successfully deployed in 2010 and the follow-up development is still ongoing.

Keywords: Information system, material management, programming, benchmarking, user interface

Sisällys

1 Johdanto 7
2 Uusmediayrityksen toimintamalli 8
3 Tutkimusongelma ja opinnäytetyön tarkoitus 9
4 Tietojärjestelmien suunnittelu ja kehittäminen 10
5 Työnjako 11
6 Käsitteet 12
6.1 Hypertext Markup Language 12
6.2 Cascading Style Sheets 12
6.3 Javascript 13
6.4 AJAX 13
6.5 jQuery 13
6.6 PHP 14
6.7 Tietokanta 14
6.8 MySQL 14
6.9 MD5 15
7 Menetelmät 15
7.1 Rational Unified Process 15
7.2 Benchmarking 16
7.3 Aivoriihi 17
7.4 Dokumentointi 18
8 Työkalut 18
9 Muiden järjestelmien tutkiminen ja vertailu 19
9.1 Käytettävyys 22
9.2 Toiminnallisuudet 24
9.3 Ulkoasu 26
10 Materiaalinhallintajärjestelmän kehittämisen vaiheet 26
10.1 Määrittely 26
10.1.1 Vaatimusmäärittely 27
10.1.2 Erityistarpeiden määrittely 28
10.1.3 Järjestelmän käyttäjäryhmät 31
10.1.4 Järjestelmän toiminnallisuudet 34
10.2 Suunnittelu 42
10.2.1 Toteutukseen käytettävät tekniikat 42
10.2.2 Rakenne 42
10.2.3 Ohjelmamoduulit 44
10.2.4 Käyttöliittymä 44
10.2.5 Tietokanta 48

10.2.6 Tietoturva 50
10.3 Toteutus 51
10.3.1 Käyttöliittymän toteuttaminen 51
10.3.2 Tietokannan ja sen taulujen rakentaminen 54
10.4 Testaus ja sen suunnittelu 56
11 Materiaalinhallintajärjestelmän jatkokehitys 57
11.1 Versionhallinta 58
11.2 Käyttöliittymämuutokset 58
11.3 Esikatselu 59
12 Pohdinta 59
Lähteet 61
Liitteet 63
Liite 1 Sisäänkirjautumista käsittelevä ohjelmakoodi 63
Liite 2 Järjestelmän päänäkymän pohjustus 64
Liite 3 Käyttöliittymätoimintoihin liittyvää ohjelmakoodia 65
Liite 4 Katkelma käyttöliittymän tyylitiedostosta 69
Liite 5 Tiedostonselaus -ikkunan sivupohja 71
Liite 6 Tietokantataulujen SQL –kyselyt 72
Liite 7 Projektiaikataulu 73

1 Johdanto

Tietojenkäsittely ja tietojärjestelmät ovat vuosi vuodelta kasvattaneet suosiotaan yritysten toiminnassa. Yleisesti pääperiaatteena uusien tietojärjestelmien käyttöönotossa on, että niillä saadaan yksinkertaistettua ja tiivistettyä yrityksen toimintoja ja prosesseja. Toisaalta tietojärjestelmät voivat myöskin olla yrityksen työkaluja, joita käyttämällä mahdollisetaan kokonaan uusia toimintoja, joita ilman järjestelmää olisi mahdoton harjoittaa.

Hyvin useasti kuitenkin tilanne on se, että yrityksellä on konkreettinen ongelma liittyen johonkin heidän prosesseistaan, ja uuden tietojärjestelmän kehittämisellä ja käyttöönotolla kyseinen prosessi joko poistuu kokonaan tai yksinkertaistuu ja nopeutuu. Näistä syistä monet yritykset ovat valmiita panostamaan suuriakin summia rahaa ja resursseja uusien järjestelmien kehittämiseen.

Suomessa mainosalan yrityksen toimintamallina on ulkoistaa tuotanto ja keskittyä itse vain mainonnansuunnitteluun ja -konseptointiin. Tämä on mahdollistanut nopean kasvun uusmediayritysten joukossa. Suuri osa näistä Suomen uusmediatoimistoista toimii läheisessä yritys-suhteessa mainostoimistojen kumppanina. Nämä uusmediatoimistot keskittyvät vain tuotannollisiin kysymyksiin ja tuovat oman lisäarvonsa mainostoimistojen projekteihin erikoistumalla erilaisiin uusiin tuotantotekniikoihin, medioihin ja niihin liittyviin innovaatioihin. Hyvin useasti nämä yritykset eivät kuitenkaan ota kantaa alkuperäisen mainos- tai uusmediaprojektin ideaan tai konseptiin, jonka asiakasyritys on jo valmiiksi suunnitellut.

Valtaosassa projekteista on hyvin useasti monia eri toimijoita, jotka toimivat yhdessä toteuttaakseen monimediaisia projekteja. Tästä syystä uusmedia-alalla on haasteita erityisesti materiaalinhallinnassa kaikkien kumppaneiden välillä. Yrityksen välittävät toisilleen suuria tiedostoja sekä salassapidettävää materiaalia, joka ei saa joutua kolmansille osapuolille. Useasti tämän materiaalin välittäminen esimerkiksi sähköpostitse tai julkisen FTP-palvelimen välityksellä on lähes mahdotonta, koska tiedostokoot ovat suuria, tiedostoja on paljon tai niitä ei voi julkaista täysin avoimella alustalla.

Näihin haasteisiin vastatakseen hyvin monet uusmediayritykset ovat turvautuneet joko Open-Source- tai kaupallisiin materiaalinhallintajärjestelmiin, jotka palvelevat heidän liikkuvaa työtä ja tiedostojenvälitystarpeitaan.

Opinnäytetyön toimeksiantajana toimi uusmediatoimisto nimeltä Barabra Oy, joka on jokapäiväisessä toiminnassaan törmännyt materiaalinhallintaongelmiin yhteistyökumppaneidensa sekä asiakkaidensa kanssa. Tästä syystä heillä oli tarve kehittää juuri omia tarpeitaan palveleva, täysin selainpohjainen materiaalinhallintajärjestelmä, joka on käytettävyydeltään

yksinkertainen, selkeä ja toimintalogiikaltaan joustava. Selainpohjainen järjestelmä mahdollistaa järjestelmän käytön sijainnista riippumatta.

2 Uusmediayrityksen toimintamalli

Uusmedia- ja digitaaliset tuotantotoimistot ovat erikoistuneet erilaisiin median tekniikoihin ja toimivat esimerkiksi mainos-, elokuva- ja mediatoimistojen yhteistyökumppaneina. Näiden yritysten projekteilla on yleensä sellaisia tuotannollisia tarpeita, joihin ne eivät oman yrityksensä sisältä löydä riittävästi tietoa ja taitoa. Myöskään tällaisen henkilöstön hankkiminen ei ole rahallisesti asiakasyritykselle kannattavaa, koska projektit ovat aina erilaisia ja useasti toteutukseen käytettävät tekniikat ovat vaihtelevia tai hyvin harvinaisia.

Uusmedian tuotantotoimistot täyttävät tämän osaamisen aukon hankkimalla esimerkiksi eri ohjelmointikielten, ääni- ja videotuotannon sekä graafisen alan ammattilaisia, jolloin ne pystyvät toteuttamaan projektin lähes kaikki tuotannon osa-alueet yrityksen sisällä. Tämä tuo loppuasiakkaalle suurta lisäarvoa projektiin erityisesti resursoinnin suhteen.

Opinnäytetyön kohdeyritys on uusmediatoimisto Barabra Oy, joka on lähivuosina törmännyt useisiin materiaalinhallinnallisiin haasteisiin jokapäiväisessä toiminnassaan. Projektien koko ja ajanjaksot kasvavat, ja näin ollen myös projektien lähdemateriaalit ja niihin käytettävät resurssit kasvavat.

Tähän asti yrityksessä on käytetty materiaalienhallintaan hyvin perinteisiä toimintatapoja. Pienikokoiset tiedostot ovat siirtyneet asiakkaille ja kumppaneille sähköpostitse, kun taas suurempikokoisia tiedostoja ja projekteja joudutaan välittämään joko FTP-palvelimen avulla tai fyysisillä medioilla kuten CD- ja DVD-levyillä.

Yrityksen sisäiset materiaalit kuten kirjasintiedostot ja raaka kuvamateriaali on pyritty säilyttämään yrityksen sisäisessä verkossa sijaitsevalla verkkolevyasemalla. Tälle verkkolevyasemalle ei kuitenkaan ole pääsyä muualta kuin toimistotilojen sisäverkosta, joten tiedostoja ei pääse käsittelemään esimerkiksi kotoa käsin. Hyvin useasti projektien osa-alueita toteutetaan esimerkiksi asiakasyritysten tiloissa, joista ei ole pääsyä Barabra Oy:n omaan sisäverkkoon. Siitä johtuen projektin aikataulut kasvavat jo pelkästään materiaalienhallinnan hitauden takia.

Edellä mainittujen tällähetkellä käytössä olevien toimintatapojen ongelmat vain kasvaneet yhteistyökumppaneiden määrän kasvaessa. Seuraavaan taulukkoon on eriteltynä tällähetkellä kohdeyrityksen käytössä olevien eri materiaalinhallinnallisten keinojen ongelmat ja heikkoudet, jotka tulivat ilmi yrityksen sisäisissä aivoriihissä.

Taulukko 1. Eri materiaalinhallintakeinojen käytössä todetut ongelmat

Keino	*Heikkoudet, riskit ja ongelmat*
Fyysiset mediat (CD / DVD)	- Hidas - Hankala varmuuskopioida - Voidaan osoittaa vain yksittäiselle henkilölle - Vaatii erilliskustannuksia (mm. lähettipalvelut)
Sähköposti	- Hidas - Ei turvallinen - Epäkäytännöllinen - Suuret tiedostot rasittaa sähköpostipalvelimia turhaan - Varmuuskopiointi / arkistointi järkevästi lähes mahdotonta
FTP -palvelimet	- Hidas - Ei turvallinen - Asiakasyrityksillä ei välttämättä mahdollisuutta käyttää - Vaatii erityisosaamista

3 Tutkimusongelma ja opinnäytetyön tarkoitus

Kuten edellä todettiin, tällä hetkellä kohdeyrityksellä ei ole käytössään minkäänlaista erillistä järjestelmää tiedosto- ja projektimateriaalien hallintaan, lukuun ottamatta verkkoasemaa, joka on alun perin suunniteltu vanhojen projektien arkistointitarpeisiin. Opinnäytetyön tavoitteena on suunnitella ja toteuttaa Barabra Oy:lle tämän verkkolevyaseman korvaava tietokantapohjainen materiaalinhallintajärjestelmä tuotantotoimiston kasvaviin tarpeisiin. Erityisenä ongelmana yrityksen sisäisen verkkoaseman käytössä on se, että sinne ei ole pääsyä kuin yrityksen työntekijöillä. Näin ollen muun muassa asiakkaat, alihankkijat ja muut sidosryhmät, esimerkiksi painotalot, eivät pääse käsiksi tarvitsemiinsa tiedostoihin. Tämä on johtanut siihen, että tiedostot ovat tähän asti levinneet sähköpostitse tai FTP-palvelinten kautta.

Nämä aiemmat materiaalintallennustavat ja -perinteet ovat johtaneet siihen, että tiedostoja ja materiaaleja on arkistoitu useaan eri paikkaan. Tämän lisäksi suurikokoisten, esimerkiksi painokelpoisten, mainosgrafiikkatiedostojen välittäminen sähköpostitse on lähes mahdotonta sekä täysin epäkäytännöllistä. Hyvin useasti projektit koostuvat myös monesta pienemmästä kokonaisuudesta kuten mainosbannereista ja www-sivustosta, ja myös tämän toimittaminen sellaisenaan sähköpostitse on hankalaa. Fyysiset mediat kuten CD- ja DVD -levyt hidastavat

materiaalin kulkua yrityksestä aina asiakkaalle, joten sekään ei ole aikataulujen puitteissa toimiva ratkaisu.

Opinnäytetyön tutkimusongelmana onkin siis tutkia millaisella palvelulla tai järjestelmällä kohdeyrityksen materiaalinhallintaongelmat saadaan ratkaistua kustannustehokkaasti. Ratkaisun tulee olla myös niin joustava, että sen lisäksi että tämänhetkinen ongelma ratkeaa, toimii se myös ratkaisumallina pitkällä aikavälillä, eli on niin kehityskykyinen että sitä voidaan tarpeen vaatiessa jatkokehittää tarpeiden vaihtuessa tai muuttuessa. Tämän tutkimuksen tuloksena kehittettävää järjestelemää käyttämällä projektien materiaalit pyritään saamaan keskitetysti yhteen järjestelmään eikä vanhoja projekteja ole arkistoituna enää useaan eri paikkaan. Kysymyksiä, joihin opinnäytetyön tutkivan osuuden sekä lopputuloksena syntyvän käytännön sovelluksen tulisi vastata, ovat:

- Millä vanhat ja ongelmalliset materiaalinhallintakeinot voidaan korvata?
- Kenelle materiaalia tulee pystyä lähettämään?
- Kenen tulee päästä käsiksi materiaaleihin?
- Miten materiaalia on järkevin hallita?

4 Tietojärjestelmien suunnittelu ja kehittäminen

Tietojärjestelmien avulla tiedonhallinta ja tarkastelu on kehittynyt merkittävästi. Järjestelmien teknologioiden kehitys avaa yritykselle runsaasti uusia mahdollisuuksia tietomäärän sekä ominaisuuksien kasvamisen myötä.

Tietojärjestelmien visualisointi, määrittely ja dokumentointi vaativat, että järjestelmää tutkitaan useasta eri näkökulmasta. Kaikki käyttäjäryhmät tulee ottaa huomioon. Analyytikot, kehittäjät, testaajat ja projektipäälliköt tuovat kaikki oman näkemyksensä projektiin.

Tarve hallita suuria datamääriä on suuri. Yritykset tallentavat jatkuvasti enenevässä määrin dataa ja yrityksen materiaaleja tietojärjestelmiin ja digitaaliseen muotoon. Datan tulee olla jatkuvasti helposti haettavissa, tallennettavissa sekä hallittavissa. Yrityksestä riippuen tarve tietojärjestelmille vaihtelee paljon. Tästä syystä tarve kehittää ja ylläpitää tekniikoita, jotka pystyvät vastaamaan jokaisen yrityksen tarpeisiin vähin kustannuksin tarpeeksi nopeasti, kasvaa. Yksi näistä tekniikoista on hajautettu tietojärjestelmä.

Koska opinnäytetyössä kehitettävä järjestelmä käsittelee ja tallentaa tietoa käyttäen useaa tietokonetta, voidaan se määritellä niin sanotuksi hajautetuksi tiedostojärjestelmäksi. Hajautetun tiedostojärjestelmän tehtävänä on tarjota verkonlaajuisesti tiedostopalvelua. Perusaja-

tuksena hajautettu tiedostojärjestelmä tarjoaa väliaikaisen tallennuspaikan tiedostoille suurikapasiteettisilla ja suuren suorituskyvyn omaavilla verkkolevyillä ja järjestelmillä. (Crichlow 2001, 20.)

5 Työnjako

Työnjako pyrittiin tekemään mahdollisimman johdonmukaisesti ottaen huomioon molempien opinnäytetyön tekijöiden parhaat osaamisalueet. Molemmat kehittäjät toimivat työnantajayrityksessä lähes samoissa työtehtävissä, mutta selkeät erot osaamisessa ja ammatillisissa intohimoissa oli kuitenkin olemassa.

Järjestelmän kehittäminen jaettiin työnjaollisesti seuraaviin pääosa-alueisiin:

- graafinen suunnittelu
- järjestelmän käyttöliittymän ohjelmointi ja rakennus
- järjestelmän toiminnallisuuden ohjelmointi
- opinnäytetyön kirjallisen osion tuotanto.

Graafisen suunnittelun toteutti Antti Sinisalo, joka on työtehtäviensä puolesta keskittynyt selkeästi työtoveriaan enemmän graafiseen suunnitteluun. Sinisalolla on myös huomattavasti enemmän kokemusta graafisesta suunnittelusta ennen työelämään siirtymistä, joten tämä osa-alue annettiin kokonaan Antti Sinisalon hoidettavaksi. Ensimmäisten suunnitelmien valmistuttua päätettiin kuitenkin katsoa työjälkeä myös yhdessä, ja pohtia mahdollisia muutoksia ja kehitysehdotuksia.

Järjestelmän ohjelmointi päätettiin jakaa kahteen osaan, koska se oli laajudeltaan hyvin suuri. Käyttöliittymän ja sivupohjien rakennus sekä itse järjestelmän toiminnallisuuden ohjelmointi päätettiin pitää erillisinä osa-alueinaan. Molemmat osa-alueet oli hyvä jakaa omalle kehittäjälleen, koska näin järjestelmän suurista kokonaisuuksista saatiin selkeä kokonaiskuva ohjelmoinnin osalta. Mahdolliset vastaantulevat ongelmatkin oli näin helpompi kartoittaa, kun tiedettiin heti kuka on kyseisen osa-alueen tehnyt ja missä mahdollinen virhe saattoi sijaita. Käyttöliittymän ja sivupohjat rakensi Esko Räsänen ja järjestelmän taustatoiminnallisuuden ohjelmoi Antti Sinisalo. Järjestys päätettiin sen perusteella, että Sinisalolla on enemmän kokemusta Ajax-tekniikasta. Koska järjestelmä tulee pohjautumaan täysin tähän tekniikkaan, on se kannattavampaa tehdä Sinisalon toimesta.

Opinnäytetyön kirjallisen osan toteuttamisen koordinoi Esko Räsänen, koska tässä vaiheessa käytönnön osuuden työmäärän koettiin kallistuvan hieman enemmän Antti Sinisalon kohdalle. Molemmat kehittäjät pidettiin hyvin perillä siitä, mitä toinen teki ja yhteistyötä tehtiin jatku-

vasti keskustelemalla ja suunnittelemalla kaikkia osa-alueita yhdessä. Näin oltiin perillä siitä mitä toinen teki ja säilytettiin selkeä kokonaiskuva projektista ja sen raportoinnista. Projektin aikataulu on esitettynä erikseen myös liitteenä löytyvästä Ganttin kaaviosta (Liite 7).

6 Käsitteet

Tämä osuus opinnäytetyön kirjallisesta raportista avaa eri tekniset termit ja tekniikat joista raportin eri vaiheissa puhutaan ja joita käytännön toteutuksessa käytettiin. Tavoitteena on auttaa lukijaa ymmärtämään tietyt perustekniikat, jolloin raportin läpikäyminen lukijan näkökulmasta helpottuu.

6.1 Hypertext Markup Language

HTML eli Hypertext Markup Language on standardoitu kuvauskieli, jolla voidaan kuvata hyperlinkkejä sisältävää tekstiä eli hypertekstiä. HTML tunnetaan erityisesti kielenä, josta websivut rakentuvat.

HTML:lla voidaan myös merkitä tekstin rakenne, eli esimerkiksi mikä osa tekstistä on otsikkoa ja mikä sisältötekstiä. Merkintä tehdään tekstin sekaan kirjoitettavilla elementeillä ja elementeissä olevilla määritelmillä.

Verrattuna muihin ohjelmointikieliin HTML on sidoksissa aikaan sekä tilaan, eli se on siis muuttumatonta. Tämä tarkoittaa, että HTML -kielellä ei pystytä sellaisenaan tekemään muutoksia sisältöön ajon aikana. (Jesper Ek & Karl-Johan Norén 1999, 1.)

6.2 Cascading Style Sheets

CSS eli Cascading Style Sheets on erityisesti www-dokumenteille kehitetty tyyliohjeiden laji. CSS on nimenomaan kaskadinen tyyliohjejärjestelmä, eli dokumenteille voi määritellä useita tyyliohjeita, jotka yhdistetään tietyllä tavalla yhdeksi säännöstöksi. (Jesper Ek & Karl-Johan Norén 1999, 4.)

Muihin tyylikieliin verrattuna CSS on hyvin yksinkertaista ja on kehitetty erityisesti soveltumaan HTML:n kanssa käytettäväksi (Jesper Ek & Karl-Johan Norén 1999, 4.). HTML:n yhteydessä käytettynä CSS-tyylitiedostossa määritellään useasti HTML-sivuston ulkoasu ja graafinen käyttöliittymä, kun taas itse HTML -tiedosto sisältää vain rakenteen ja sisällön.

6.3 Javascript

Javascript on Netscape Communications Corporationin kehittämä pääasiassa web-ympäristössä käytettävä ohjelmointikieli. Javascriptin tärkein sovellus on mahdollisuus lisätä web-sivuille dynaamista toiminnallisuutta, jota HTML itsessään ei sisällä.

Javascriptin nykymuoto on oliopohjainen kieli, jonka syntaksi perustuu etäisesti C-ohjelmointikieleen. Jos sivustosta halutaan interaktiivinen tai dynaaminen, tarvitaan tähän sopiva skriptikieli, kuten Javascript. (Jesper Ek & Karl-Johan Norén 1999, 7.)

6.4 AJAX

Ajax eli Asynchronous JavaScrips and XML on ryhmä web-sovelluskehityksessä käytetyistä tekniikoista, joiden avulla web-sovelluksista ja -sivustoista voi tehdä interaktiivisimpia ja dynaamisempia. Alkuperäisessä merkityksessään AJAX:lla on viitattu web-tekniikkaan, jossa verkkosivulla Javascript:llä asynkronisesti tehtävistä http-pyynnöistä voidaan palauttaa XML-merkkausta.

Nykyisin AJAX-tekniikoilla viitataan yleisesti samankaltaiseen toimintatapaan. Siinä selainohjelma välittää pieniä määriä dataa palvelimen kanssa web-sovelluksen taustalla, ja näin ollen koko näkymää ei tarvitse ladata uudelleen joka kerta kun käyttäjä tekee toimintoja käyttöliittymässä. Tällä tavoitellaan verkkopalvelun vuorovaikutteisuuden, nopeuden ja käytettävyyden lisäämistä ja parantamista.

Tällä hetkellä Ajax on yksi merkittävimmistä vaihtoehdoista monipuolisten verkkosovellusten kehittämiseen. Vaikka Javascript onkin AJAX:n keskeinen komponentti, on AJAX enemmänkin tekniikka kuin teknologia. AJAX toimii lähes kaikissa nykyaikaisissa selaimissa ilman mitään lisäohjelmistoja. Sen vahvuus on juuri se, että se rakentuu jo olemassaolevien tekniikoiden päälle. Yksi Ajaxin tunnetuimmistä hyödyntäjistä on kuuluisa hakukone Google, ja se hyödyntää tekniikkaa muun muassa Google Maps:n ja Gmailin yhteydessä. (Ryan Asleson & Nathanael T. Scutta 2006, 14.)

6.5 jQuery

jQuery on AJAX -tekniikkaa hyväksikäyttävä laaja OpenSource JavaScript -kirjasto, joka yksinkertaistaa useiden eri AJAX:n peruskäyttötarkoituksen toiminnallisuuksia. jQuery -kirjastoa on kehitetty aina vuodesta 2005 lähtien, kun henkilö nimeltä John Resig sai idean hyväksikäyttää CSS -valitsijoita yksinkertaisemmalla tavalla kuin aiemmat kirjastot. Ensimmäinen virallinen, julkinen versio jQuerystä julkaistiin kuitenkin vasta 14. tammikuuta 2006. Käyttämällä jQue-

ryä voidaan yksinkertaistaa tapaa kirjoittaa JavaScriptiä, animoida CSS -elementtejä, hyväksikäyttää tapahtumankäsittelyä sekä yleisesti nopeuttaa verkkopalveluiden interaktiivisten osioiden kehitystyötä. (jQuery Project 2009.)

6.6 PHP

PHP (lyhenne sanoista PHP: Hypertext Processor) on Perlin kaltainen ohjelmointikieli, jota käytetään erityisesti Web-palvelinympäristöissä dynaamisten web-sivujen luonnissa. Ohjelmointikielen lisäksi PHP-ympäristössä on laaja luokkakirjasto. PHP on komentosarjakieli, jossa ohjelmakoodi tulkitaan vasta ohjelman suoritusvaiheessa. PHP:tä voidaan käyttää useilla eri alustoilla ja käyttöjärjestelmillä. (Ryan Asleson & Nathanael T. Scutta 2006, 6.)

Ensimmäinen versio julkaistiin vuonna 1995, ja nykyisin PHP on vertailussa johtava dynaamisten web-palveluiden tuottamiseen tarkoitettu kieli. PHP mahdollistaa dynaamisen sisällön palvelimella ja sen palauttamisen staattisessa muodossa verkkosovellukselle. (Ryan Asleson & Nathanael T. Scutta 2006, 6.)

6.7 Tietokanta

Tietokanta on tiedonsäilytystapa, jolla tieto pystytään tallentamaan järjestäytyneesti. Tietokannan luomisen jälkeen siellä sijaitsevan tiedon etsiminen ja tarvittaessa muokkaaminen on hyvin helppoa.

Tietokannasta haetaan tietoa käyttämällä tarpeeksi rajattuja kyselyitä ja hakutoimintoja. Tietokannat ovat hyvä tapa strukturoida tietoa ja tehdä siitä nopeasti haettavaa. Esimerkiksi websivuston ja tietokannan yhdistämisen ansiosta pystytään laatimaan interaktiivisia sivustoja ja luomaan järjestelmiä. (Stefan Arvidsson & Jesper Ek 1999, 2.)

6.8 MySQL

MySQL on suosittu ja tehokas SQL-tietokannan hallintajärjestelmä, joka on asennettu yli kuuteen miljoonaan tietokoneeseen. MySQL:ää kehittää ruotsalainen yritys MySQL AB, jonka osti 16. tammikuuta 2008 Sun Microsystems. MySQL on saatavissa vapaalla GNU GPL –lisenssillä tai kaupallisella lisenssillä, jos GPL ei ole sopiva. (MySQL 2009.)

Monista kaupallisista tietokantajärjestelmistä poiketen MySQL:n hallinnointi tapahtuu komentoriviltä tai tekstipohjaisella asiakasohjelmalla. Sille on tosin valmistajan sivulta saatava graafiset MySql Administrator ja MySQL Query Browser, sekä suosittu vaihtoehtoinen phpMyAdmin.

6.9 MD5

MD5 on niin kutsuttu message-digest-algoritmi, jota käytetään muun muassa kryptografiassa. MD5 on yksi monista Ronald Rivestin kehittämistä viestitiivistealgoritmeista. Se perustuu aikaisempaan MD4-algoritmiin, jonka analyysit osoittivat mahdollisesti epäturvalliseksi. Matemaattisesti merkkijonoja, jotka tuottavat saman tiivisteen, on rajattomasti. Niiden löytäminen vain on erittäin työlästä. (What is MD5? 2009.)

MD5-algoritmi tuottaa tuloksenaan 128-bittisen tiivisteen, joka tyypillisesti esitetään 32-merkkisenä heksakoodatussa muodossaan (What is MD5?, 2009). Esimerkiksi merkkijonon *The quick brown fox jumped over the lazy dogs* MD5-tiiviste on 8c1788205b6abffb0c6a4a1f4b10395a.

7 Menetelmät

Opinnäytetyö koostuu käytännön sovelluksen toteuttamisesta sekä tiedon keräämisestä ja sen analysoimisesta. Molempiin kokonaisuuksia tukemaan valittiin tiettyjä menetelmiä ja työkaluja joita hyödynnettiin soveltamalla niitä tarkoituksenmukaisesti. Seuraavaksi on esiteltynä eri menetelmät joita opinnäytetyön eri vaiheissa hyödynnettiin.

7.1 Rational Unified Process

Yhdessä kohdeyrityksen kanssa päädyttiin käyttämään järjestelmän kehityksessä Rational Unified Process:n mukaista prosessimalli, jonka perusperiaate rakentuu määrittelystä, suunnittelusta sekä toteutuksesta (IBM 2009). RUP sopii hyvin tietojärjestelmäprojekteihin ja tämä toimintamalli oli opinnäytetyön tekijöille jo entuudestaan tuttu. RUP:n mukaiseen prosessimalliin päädyttiin siksi että siihen on liitoksissa selkeästi UML–kaaviot jotka ovat teknisen kehitystyön näkökulmasta tärkeitä. Lisäksi verrattuna esimerkiksi perinteiseen vesiputousmalliin RUP on joustavampi, joka oli myös yksi valintakriteereistä.

Ilman ennalta määrättyä suunnittelu- ja tuotantoprosessia kehitystiimi kehittää sovellusta useasti kokeilun kautta, ja onnistuminen on useasti yksilöiden suoritusten varassa. Rational Unified Process on järjestelmäkehitystä varten kehitetty prosessimalli. Se määrittelee tarkasti tehtävät ja vastuualueet kehitysorganisaation sisällä. Sen tavoitteena on varmistaa korkealaatuisen ohjelmistokehityksen tuotanto ja pitää huolta siitä, että se vastaa loppukäyttäjän tarpeita. (Kruchten 2003, 20.)

RUP pitää sisällään useita hyväksi todettuja modernin ohjelmistokehityksen käytäntöjä, ja on siten hyvin soveltuva erilaisiin projekteihin ja organisaatioihin. Se sisältää muun muassa käytännöt seuraaville asioille:

1. ohjelmistokehitys vaiheittain
2. vaatimustenmäärittely ja hallinta
3. komponenttipohjainen arkkitehtuuri
4. visuaalinen ohjelmiston mallintaminen.

(Kruchten 2003, 20.)

Erityisesti opinnäytetyön yhteydessä hyödynnettiin RUP:ia siten että projekti jaettiin selkeästi eri vaiheisiin. Projektisuunnitelmien ja muiden opinnäytetyöprosessiin kuuluvien dokumenttien totetuttamisen jälkeen projektin käytännön sovelluksen suunnittelu ja toteuttaminen jaettiin selkeästi erillisiin määrittely-, suunnittelu- ja toteutusvaiheisiin. Tällöin projektista saatiin talteen rakenteeltaan jäsennelty dokumentaatio opinnäyteyöraportin muodossa sekä tarvittavat UML–kaaviot ja muut suunnitteluun hyvin keskeisesti liittyvät tekniset dokumentit kuten esimerkiksi käyttötapauskaaviot.

7.2 Benchmarking

Yhtenä tiedonkeruun menetelemänä itse kehitystyön yhteydessä käytettiin benchmarkingia. Ennen käytännön toteutuksen kehitystyötä, useita vastaavia jo olemassa olevia materiaalinhallintajärjestelmiä tutkittiin käyttäen apuna ohjelmistotuotteiden laatustandardia ISO 9126. Näin saatiin mahdollisimman käyttökelpoisia tuloksia, joita hyödyntämällä kehitettävän järjestelmän määrittely ja suunnittelu helpottui. ISO 9126 laatustandardissa on jaettu ohjelmistolaatu kuuteen eri pääominaisuuteen, jotka ovat functionality, reliability, usability, effiency, maintainability ja portability. (Talentum 2005, 92-93.)

Lisäksi laatustandardin määrittelemien pääominaisuuksien alle lisättiin kohdeyrityksen kanssa aivoriihissä läpikäydyt määrittelyt ja vaatimukset, joiden perusteella verrattiin myös valmiita ohjelmia ristiin omien kokemuksien sekä aiemmin kehitettyjen järjestelmien kanssa. Pyrimme näin löytämään omien sekä muiden järjestelmien heikkoudet ja vahvuudet, jotta voisimme oppia rakentamaan parhaan mahdollisen järjestelmän kohdeyrityksen vaatimusmäärittelyn sekä laatustandardin pohjalta.

Tämän tarkoituksena oli välttää mahdollisia ongelmia tai kehittää niihin toimivat ratkaisut. Tutkimalla useita vastaavia järjestelmia sekä soveltamalla niitä ja tähän asti opittua tietoutta pyrittiin löytämään oikeat ratkaisut kunkin ongelman ratkaisemiseen. Yhdistämällä eri oh-

jelmien parhaita ominaisuuksia ja kehittelemällä niiden heikoista ratkaisuista toimivia, saatiin myös hyvin käyttökelpoista tietoutta siitä mitä kannattaa omassa järjestelmässä toteuttaa ja miten. Tarkoituksena oli siis kehittää jotain täysin uutta, joka soveltuu juuri kohdeyrityksen ratkaisuksi.

Tutkimukseen valitut ohjelmat valittiin käyttäen harkinnanvaraista otosta. Harkinnanvarainen otos sopii tutkimuksiin, joissa ei pyritä tulosten yleistämiseen suurempaan perusjoukkoon (Vilkka 2007, 58). Vertaillut ohjelmat ja tulosten analysoinnin tulokset ovat esiteltynä myöhemmässä vaiheessa tätä raporttia.

7.3 Aivoriihi

Aivoriihi (englanniksi ”brainstorming”) on luovan ongelmanratkaisun standardimenetelmä, jonka tavoitteena on saada aikaan ja kehittää suuri määrä luovia ideoita siten, että kaikki ryhmän jäsenet osallistuvat ideointiin. Aivoriihen periaatteena on, että suuri ideoiden määrä tuottaa laatua. Mitä enemmän ideoita on, sitä todennäköisemmin osa niistä on toteauttamiskolpoisia. (Helsingin Yliopisto, 2010.)

Ennen käytännön kehitystyön aloittamista yhdessä kohdeyrityksen henkilöstön kanssa käytiin läpi järjestelmän perustoiminnallisuuksien toteuttamista sekä ideoitiin yhdessä mahdollisia uusia alkuperäisten suunnitelmien ulkopuolisia toiminnallisuuksia. Pääasiallisesti aivoriihet toteutettiin ryhmässä johon kuului seuraavat henkilöt:

- Esko Räsänen
- Antti Sinisalo
- Jari Pelto-Piri
- 2 Barabran ohjelmistokehittäjää

Pääasiassa aivoriiheä hyödynnettiin projektin suunnitteluvaiheessa. Yrityksen sisällä käydyissä aivoriihissä saatiin hyvin paljon ideoita erityisesti kehitettävän järjestelmän käyttöliittymään liittyen. Myöhemmin projektin edetessä suunnittelun pääteemaksi valittiin käyttäjälähtöisyys ja yksi suurin aivoriihien tuottama idea olikin käyttöliittymän muokkaaminen käyttäjäkohtaisesti. Lisäksi aivoriihet oli hyödyksi järjestelmän teknisten ratkaisujen valinnassa, koska mukana oli kaksi hyvin asiantuntevaa ohjelmistokehittäjää.

Käyttöliittymän muokattavuuden lisäksi aivoriihissä käytiin läpi järjestelmältä vaadittavat perustoiminnallisuudet kuten tiedostojen lisääminen, poistaminen ja muokkaaminen sekä käyttäjänhallinta. Tuloksena näistä aivoriihistä saatiin siis kattava kokoelma erilaisia ideoita

ja toiveita siitä miten kehitettävän järjestelmän tulisi toimia ja miltä sen tulisi pääpirteiltään näyttää.

7.4 Dokumentointi

Työn eri vaiheiden selkeän erittelyn lisäksi tärkeää oli tarkka ja yksityiskohtainen dokumentointi. Tähän opinnäytetyöhön on liitetty osa ensimmäisen toimivan version suunnittelu- ja määrittelydokumentaatioista sekä erilaisia kaavioita. Hyvin useasti tietojärjestelmät ja niiden tietokannat tulevat vaatimaan muutoksia yrityksen muuttuneiden tarpeiden mukaisesti. Tästä syystä työn ja sen tulosten tarkka dokumentointi ja seuraaminen on hyvin tärkeä osa tietojärjestelmäkehitystä. Barabran kanssa yhteistyössä käytiin läpi erikseen, mitä dokumentteja järjestelmästä tarvitaan ja miten eri asiat tulee kuvata.

Koska järjestelmä suunniteltiin ja toteutettiin RUP-prosessin mukaisesti, sivutuotoksena tuotettu dokumentointi sisältää myös useita eri kaavioita ja määrittelytaulukoita. Lisäksi tietokannoista jäi kattavat dokumentaatiot käytettyjen tietokantaohjelmistojen myötä. Koska tietokanta rakennettiin ja suunniteltiin käyttäen erityisesti siihen tarkoitukseen tehtyä ohjelmistoa, tietokannoista tallentui suoraan dokumentit.

Lisäksi järjestelmän lähdekoodi kommentointiin niin, että järjestelmän jatkokehitys tulisi olla erittäin helppoa. Vaikka järjestelmää tulisi jatkossa kehittämään henkilö, joka ei ole ollut järjestelmän alkuperäisessä kehitysprojektissa mukana, tulisi hänen olla mahdollista seurata ohjelmakoodia ja sen tietovirtaa ongelmitta. Lähdekoodin selkeä kommentointi auttaa myös kehittäjiä järjestelmän tuotantovaiheessa, sekä lisäksi se toimii yhtenä dokumentointimuotona. Dokumentointia voidaankin näin pitää yhtenä opinnäytetyöprojektin menetelmistä, koska se osaltaan tuki projektin selkeästi jäsenneltyä läpivientiä.

8 Työkalut

Barabra Oy:lla oli käytössään laaja valikoima suunnitteluun ja toteutukseen liittyviä työkaluohjelmistoja, joita käytettiin luomaan parhaat mahdolliset puitteet uuden järjestelmän toteuttamiselle. Keskusteluissa kohdeyrityksen kanssa ei nähty tarpeelliseksi tehdä uusia ohjelmistohankintoja projektia varten. Tarkoituksena oli saada aikaan järjestelmä ilman erillisiä kustannuksia. Tästä samasta syystä käytettiin myös ilmaisia Opens Source-ohjelmia tarpeen näin vaatiessa.

Onnistuneen järjestelmätuotannon edellytyksenä pidettiin, että käyttöön saatiin oikeanlaiset ohjelmistot ja kehitysympäristöt niin mallintamiseen, käyttöliittymäsuunnitteluun, tietokan-

takehitykseen kuin ohjelmointiinkin. Tuttuja työkaluja käyttäen päästiin myös parhaaseen mahdolliseen lopputulokseen, koska itse työkalujen käytön opetteluun ei kulunut aikaa.

Opinnäytetyön käytännön sovelluksen suunnittelussa ja toteutuksessa käytettiin näin ollen hyvin paljon erilaisia työkaluohjelmistoja, jotka ovat erikseen esiteltynä raportin myöhemmissä osioissa.

9 Muiden järjestelmien tutkiminen ja vertailu

Kuten aiemmin todettiin, yhtenä tiedonkeräämisen menetelmänä opinnäytetyöprosessin yhteydessä käytettiin benchmarking –menetelmää. Vertailtavaksi valittiin joitain vastaavaan tarkoitukseen suunniteltuja sovelluksia.

Ensimmäiseksi tutkittavaksi ohjelmistoksi valittiin Dolphin PHP Filemanager, joka tarjoaa yksinkertaisen ja helpon ratkaisun tiedostojen tallentamiseen ja muokkaamiseen palvelimella selaimen välityksellä. Erityisen Dolphinista tekee se, että siitä on tarjolla kaksi erillistä versiota: ”Full” sekä ”Lite” -versiot.

Versiot eivät erotu toisistaan ainoastaan ominaisuuksiensa vuoksi, vaan ”Lite” –versio on tarkoitettu selattavaksi kännykällä sekä vanhemmilla selaimilla, ja “Full” -versio on toteutettu hyödyntäen Web 2.0 -tekniikkaa ja se toimii näin ainoastaan uusimmilla selaimilla. Ominaisuuksiensa puolesta Dolphin on hieman kehittyneempi sovellus verrattuna muihin tutkittuihin järjestelmiin, mutta ulkoasunsa puolesta se on hyvin lattea eikä herättänyt käyttäjässä suurta mielenkiintoa. Toiminnallisuuksiensa puolesta se kuitenkin toimi ongelmitta ja järjestelmä sisältää myös käyttäjienhallinnan.

Dolphinin ulkoasu (“Full” -versio) oli tehty täysin mukaillen Windows-käyttöjärjestelmän käyttöliittymää, joten se on tuttu Windows-käyttäjille. Toiminnot sekä valitun tiedoston tiedot sijaitsevat vasemmassa palstassa ja oikealla palstalla navigoidaan tiedostojen kesken. Käyttöliittymän mukauttaminen vastaamaan täysin Windowsin käyttöliittymää toimii hyvin, mikäli kaikki käyttäjät käyttävät jokapäiväisesti Windowsia, mutta jos käyttäjä on tottunut käyttämään esimerkiksi Macintosh–käyttöjärjestelmää, voi käyttöliittymä tuntua hyvin epäloogiselta. Toiminnallisuuksien osalta Dolphin sisältää samat perustoiminnot kuin kaikki muutkin vastaavat ohjelmistot, mutta näiden lisäksi sen mukana tulee lähes täydellinen mobiiliversio, sekä komentoja voi syöttää selaimen lisäksi komentoriviltä.

Edellä mainitun lisäksi vertailuun valittiin WebInsta Filemanager, joka on yksi tuote WebInstan PHP -pohjaisesta tuoteperheestä, johon kuuluu myös postituslista- sekä sisällönhallintaohjelmisto. Webinsta Filemanager on niin ikään luotu helpoksi asentaa sekä käyttää. Siitä löytyy valmiina kaikki perustoiminnot muttei juuri enempää. Suurin ero muihin vastaaviin ilmaisiin

sovelluksiin oli sen ulkoasu, joka erottui edukseen. Se teki sovelluksesta paljon uskottavamman ja miellyttävämmän käyttää. Käyttöliittymässä ei ole pyritty siihen, että sovellus näyttäisi esimerkiksi Microsoftin ohjelmalta. Käyttäjälähtöisyys mielessä pitäen, ulkoasu vaikutti oli yksi tärkeimmistä asioista, joihin päätettiin panostaa myös omassa järjestelmässämme.

WebInstan Filemanagerissa on ulkoasun lisäksi panostettu myös käytettävyyteen. Toiminnot on esitetty havainnollistavilla ja selkeillä ikoneilla, jotka ovat näkyvästi esillä, eikä ruudulla ole kerrallaan liikaa tavaraa. Kaikki tarvittava on jäsennelty ruudulle tyylikkäästi harkiten.

Edellämainittujen lisäksi vastapainoksi valittiin käyttöliittymältään hieman erilainen Ajax Filemanager. Suurimmaksi ongelmaksi Ajax Filemanagerin käytössä todettiin täydellisen käyttäjähallinan puute.

Sovellus vaikuttaa melko keskeneräiseltä niin ulkoasunsa kuin käytettävyytensä osalta. Ikoneina on käytetty Windowsista tuttuja ikoneita, ja kokonaisuus vaikutti ensivaikutelmaltaan hyvin epäselvältä. Ohjelma toimi kuitenkin moitteettomasti ja osoittautui ominaisuuksiensa osalta hyvinkin kelvolliseksi. Siinä on selkeä idea mutta se ei välity käyttäjälle heti ensi käyttökerroilla. Selaimeen ilmestyy heti sisäänkirjautumisen jälkeen eräänlainen työpöytä, joka on jaettu kahteen erilliseen osioon. Toisessa osiossa tapahtuu kaikki toiminnallisuus ja toinen osio toimii tavallaan eräänlaisena infotauluna, missä näkyy kaikki tiedot tiedostosta tai kansiosta, jota sillä hetkellä selataan.

Taulukossa 4 on esitetty tutkimuksen käyttökokemusten tuloksena annetut arvosanat. Taulukko sisältää pääominaisuuksien painokertoimet sekä arvosanat, jotka on laskettu keskiarvoa hyväksi käyttäen. Ennen kuin lähdettiin tutkimaan muita järjestelmiä, päätettiin yrityksen sisällä erilaisia kriteereitä, joihin laatustandardin määrittämien pääominaisuuksien lisäksi kiinnitettiäisiin erityisesti huomiota. Nämä kriteerit käyvät ilmi taulukosta.

Taulukko 2. Vertailussa mukana olleiden ohjelmien arvosanat toiminnallisuuksittain

Arvosanat asteikolla 0-10	*Painoarvo*	*Dolphin*	*PhpFilemanager*	*AjaxFilemanager*	*WebInsta*
Toiminnallisuus	41%				
Tiedoston lataaminen		10	7	8	10
Tiedoston poistaminen		10	7	8	10
Tiedostojen selaaminen		8	8	7	9
Tiedoston lähettäminen		0	0	0	0
Tiedostojen muokkaaminen		8	5	9	9
Käyttäjänhallinta		0	0	0	9
Käyttäjien lisääminen		0	0	0	9
Käyttäjätietojen muokkaaminen		0	0	0	9
Käyttäjätunnusten poistaminen		0	0	0	9
Luotettavuus	4%				
Virheetön toiminta		10	5	10	10
Käytettävyys	18%				
Navigointi		9	5	7	10
Selkeys		8	6	7	9
Käyttäjän opastaminen		8	4	5	9
Toimintalogiikka		9	6	8	9
Tuotannollinen tehokkuus	14%				
Ajankulutus		8	6	8	9
Resurssien kulutus		8	5	8	9
Mukautuvuus tehokkuuteen		9	5	8	10
Siirrettävyys	23%				
Sopeutuvuus		8	7	8	9
Asennettavuus		9	6	9	10
Päivittäminen		9	8	6	9
Rinnakkaiselo		7	8	7	10
Mukautuvuus siirrettävyyteen		7	8	7	9
Keskiarvo	100%	6,7	4,8	5,9	8,9
Keskiarvo mikäli käyttäjänhallinta jätetään ottamatta huomioon	100%	6,7	4,8	5,9	7,2

Parhaan yleisarvosanan sai tiedostojenhallintajärjestelmä WebInstaFilemanager. Tulokseen vaikuttaa korostavasti WebInstan sisältämä käyttäjänhallinta, jota muissa järjestelmissä ei ollut. Järjestelmä saa kuitenkin paremmat pisteet kuin mikään muu järjestelmä, vaikka käyttäjienhallintaa ei laskettaisikaan mukaan.

9.1 Käytettävyys

Käytettävyydellä tarkoitetaan tässä tapauksessa helppokäyttöisyyttä, käyttöliittymän ymmärrettävyyttä sekä sen sisäistettävyyttä. Koska suurin osa kohdeyrityksen asiakkaista ei ole käyttänyt tietokonetta muuten kuin pintapuolisesti, on myös se otettava suunnittelussa huomioon.

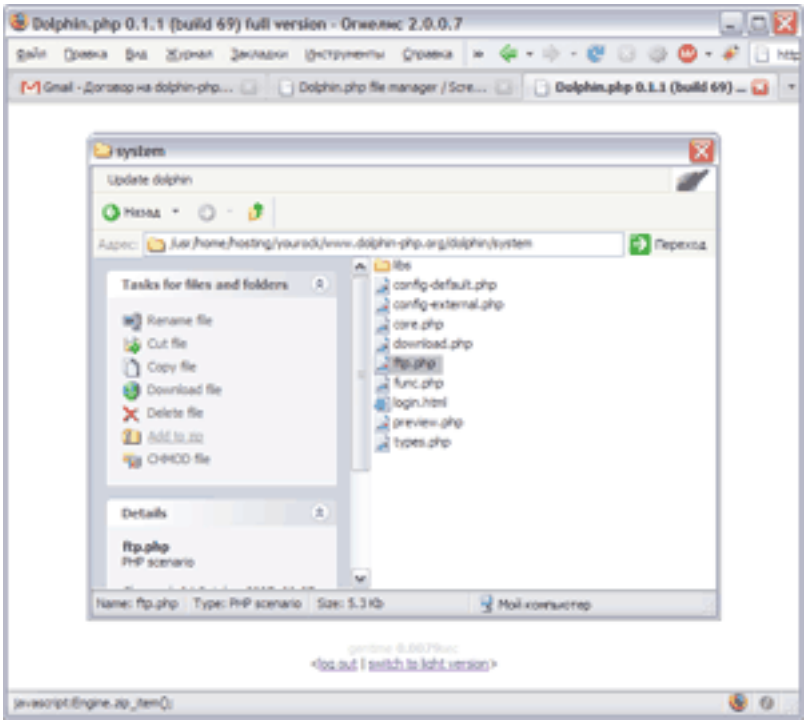
Käytettävyyden kannalta tärkeitä asioita ovat:

- Opittavuus – kuinka helppoa järjestelmän oppiminen on
- Tehokkuus – oppimisen jälkeen, sillä halutaan saavuttaa enemmän
- Muistettavuus – kuinka helppoa järjestelmän käyttäminen on oppimisen jälkeen
- Virheettömyys – käyttäjää ei saa johtaa harhaan
- Miellyttävyys – sovelluksen käytön pitää olla miellyttävää.

(Parkkinen 2002, 17.)

Käytettävyyttä koskevien kriteereiden perusteella parhaiten esiin nousi WebInsta-sovellus, joka on yksinkertainen ja selkeä. Toiminnallisuudet oli esitetty sekä ikonein että sanallisesti, sekä selkeämmät ominaisuudet ovat esitetty pelkästään kuvin ja painikkeiden ominaisuudessa. WebInsta koettiin helpoksi omaksua sekä tehokkaaksi käyttää. Joitain pieniä asioita huomattiin mitkä olivat toimintojen asettelun osalta hieman kyseenalaisia, mutta nekin omaksui nopeasti lyhyen käytön jälkeen, joten pidempiaikaisessa käytössä näitä asioita tuskin koettaisiin oikeiksi ongelmiksi. Yleisesti ohjelma on toimiva sekä käytettävyydeltään hyvin viimeistellyn oloinen.

DolphinPhp oli toteutettu täysin mukaillen Windows-käyttöjärjestelmää sekä sen käyttöliittymää. Se on graafisesti sekä toiminnallisesti tarkoitettu näyttämään täysin Windowsin kansioiden sekä tiedostojen selaukseen tarkoitettua resurssienhallintaa. Tottuneellekin Windows-käyttäjälle käy kuitenkin nopeasti ilmi, ettei toimintoja suoriteta painamalla hiiren oikeaa näppäintä tiedoston päällä, vaan kaikki toiminnot sijaitsevat ikkunan vasemmalla puolella (Kuva 1). On ymmärrettää, että totuttuja ikoneita sekä graafisia ilmeitä käytetään, mutta niihin liittyy myös selvät riskit ja ennakko-odotukset. Windows-käyttäjä saattaa hyvinkin olettaa, että voi tehdä kaikki samat tiedosto- ja kansiotoiminnot, joita on tottunut oman tietokoneensa normaalikäytössä käyttämään, koska käyttöliittymä sekä ulkoasu näyttävät täysin samalta.



Kuva 1. DolphinPhp:n tiedostojenselausnäkymä

Vertailujen tuloksena käytettävyyden osalta käytettiin hyödyksi enimmäkseen WebInstan sekä AjaxFilemanagerin hyvin toteutettuja osia sekä pohdittiin muiden ohjelmien käyttössä ilmenneitä ongelmia. Suurimmiksi ongelmiksi koettiin muun muassa turhien ominaisuuksien runsas määrä, epäselvät käyttöliittymät ja ulkoasun epäyhtenäisyys.

AjaxFilemanagerista saatiin idea luoda selkeästi erillään olevista elementeistä koostuva käyttöliittymä. AjaxFilemanagerissa oli selkeästi pohdittu tätä ominaisuutta erottelemalla toiminnallisuus ja eräänlainen infotaulu täysin erillisiksi osiksi, joka osaltaan loi todella selkeän kokonaisuuden. Infotaulu sisälsi lukuisia tietoja valitusta kansiosta ja tiedostosta. Tuota ominaisuutta ei kuitenkaan oltu hyödynnetty aivan loppuun asti, joten ominaisuutta päätettiin viedä vielä eteenpäin omassa järjestelmässä. Idea on lähes sama kuin käytetyimmissä käyttöjärjestelmissä missä ohjelmat avautuvat uusiin ikkunoihin, jolloin ne erottuvat toisistaan selkeästi.

Kaikkia ohjelmia vaivasi yksi ja sama ongelma. Kaikki tiedostot olivat listattuna allekkain ja jokaisen tiedoston yhteydessä oli kaikki toiminnallisuudet peräkkäin, kuten: muokkaa, esikatsele, ja poista. Tämäntyylinen listaus ei ole ollenkaan huono, mutta tavallaan on turhaa näyttää samoja toimintoja moneen kertaan. Toimintojen löytyessä jokaisen tiedoston kohdalta ja aina samasta kohtaa voi käyttöliittymästä saada toimivan kuvan, mutta ongelmaksi koituu mahdollisuus, jossa käyttäjä saattaa painaa toimintoa väärän rivin kohdalla. Käyttöliittymää voi kuitenkin selkeyttää sijoittamalla kaikki toiminnallisuuset yhteen erilliseen elementtiin, josta ne löytyvät aina. Tällöin jokaisen tiedoston viereen ei tarvitse erikseen sijoittaa lukuisia

ikoneita ja painikkeita, vaan toiminnot löytyvät yhdestä kohtaa ja niitä voi käyttää kaikkiin haluamiinsa tiedostoihin. Tämä nopeuttaa myös sivun latautumista, kun ladattavaa dataa on paljon. Tämän pohjalta kehitettäävän järjestelmään päätettiin rakentaa esimerkiksi yleinen roskakori tiedostojen ja kansioiden poistamista varten.

Yllämainittujen päätelmien perusteella järjestelmän käyttöliittymän eri osa-alueet päätettiin sijoittaa täysin erilleen toisistaan, selkeästi ja havainnollisesti ajatellen käytettävyyttä. Tässä vaiheessa syntyi idea innovatiivisesta käyttöliittymästä, jossa käyttäjä voisi mukauttaa näkymän itselleen sopivaksi, liikuttelemalla käyttöliittymän eri elementtejä haluamaansa järjestykseen. Näin kukin käyttäjä voisi luoda käyttöliittymästä itsellensä mielekkään, toimivan ja juuri omanlaisensa.

Lisäksi opinnäytetyössä kehitettävän järjestelmän suunnittelussa otettiin huomioon se, että enemmistö asiakkaista käyttää työtehtävissään Macintoshia. Enemmän tietokoneita käyttäneen on helpompi omaksua erilaisia käyttöliittymiä, koska niistä on enemmän kokemusta, mutta esimerkiksi vain yhtä käyttöjärjestelmää käyttävien on hyvin vaikea omaksua toinen toimintamalli ja käyttöliittymä. Asiakkaiden mieltymykset käyttöliittymien osalta ovat hyvin usein pohjautuneet heille totuttuihin käytäntöihin. Käytettävyyden ongelmana melkein jokaisessa testatussa ohjelmassa oli tarpeettomien toimintojen sekä tietojen näkyminen käyttöliittymässä. Tämä sekoittaa yleiskäytettävyyttä ja toiminnallisuutta huomattavasti. Lisäksi suurin osa testatuista materiaalinhallintaohjelmistoista käytti Windows-käyttöjärjestelmän ikoneita toiminnallisuuksiaan kuvatessa. Suurin osa näistä oli kuitenkin selkeitä, vaikka osassa ikonit saattoi kokea enemmänkin ongelmaksi.

9.2 Toiminnallisuudet

Lähes jokaisesta testatusta ohjelmistosta löytyi kaikki ominaisuudet mitä tarvittiin, poislukien käyttäjienhallinta, mahdollisuus ladata tiedostoja ilman käyttäjätunnuksia sekä tiedoston latauslinkin lähetys erikseen määriteltyyn sähköpostiosoitteeseen. Ominaisuudet itsessään olivat melko hyvin selvillä ennen eri ohjelmien tutkimista, mutta silti muutaman uusi idea saatiin vasta ohjelmia testattaessa.

Vaikka joissain ohjelmissa olikin käyttäjienhallinta, niin ohjelmat tuntuivat siltä, että ne olisi tarkoitettu käytettäväksi vain yhdellä käyttäjällä. Esimerkiksi missään ohjelmassa ei näkynyt että järjestelmään oli luotu uusia tiedostoja tai kansioita, vaan nämä asiat tuli saada selville jotain muuta kautta. Tämä ongelma koettiin melko suureksi, koska kohdeyritykselle kehitettävää järjestelmää tulisi käyttämään todella suuri määrä eri henkilöitä. Ratkaisuna järjestelmään päätettiin toteuttaa eräänlainen ”ilmoitustaulu”, joka tulisi näyttämään uusimmat ladatut tiedostot ja luodut kansiot erillisessä listassa. Listassa näkyisi tiedostojen ja kansioiden nimet, niiden lisääjät, sekä perustiedot. Tämän listaksi tiedostoihin pääsisi suoraan käsiksi

klikkaamalla jotain tiedostoa tai kansiota ilman, että täytyy erikseen selata hakemistorakennetta syvemmälle.

Järjestelmän suuren käyttäjämäärän takia päätettiin myös rakentaa eräänlainen keskustelupalsta, jossa käyttäjät voisivat lähettää toisilleen viestejä. Yhteydenpitoa varten käytetään yleisesti sähköpostia, mutta joitakin järjestelmän sisäisiä asioita ja keskusteluja voitaisiin näin käydä myös järjestelmän sisällä. Palstan kautta käyttäjä voisi vaikka antaa palautetta järjestelmästä.

Muutamissa ohjelmissa oli todella hyödyllinen esikatseluominaisuus, jonka avulla pystyi esikatselemaan kuvia erillisessä pienessä ikkunassa. Esikatselu päätettiin ensimmäisestä versiosta vielä jättää pois, mutta se testikäytössä saadun palaautteen perusteella se on tarkoitus kehittää järjestelmään jatkokehityksen yhteydessä.

Testatuissa ohjelmissa ei ollut esimerkiksi Flash –tiedostojen esikatselumahdollisuutta, mikä olisi yksi suurin käyttötarve esikatseluominaisuudelle kehitettävässä järjestelmässä. Tiedostoista suuri osa on joko Flash –tekniikalla valmistettuja mainoksia tai videoita. Päädyttiin siihen, että kun esikatselu integroidaan järjestelmään myöhemmässä vaiheessa, siinä tulisi olla myös perustiedostojen esikatselun lisäksi mahdollisuus esikatsella Flash –tiedostoja.

Esikatselu oli rakennettu eri ohjelmissa eri tavalla. Joissakin ohjelmissa oli erillinen painike, josta valitun tiedoston esikatselukuvaa pääsi tarkastelemaan erilliseen ruutuun, mutta hyödyllisemmäksi koettiin käytettävyydenkin osalta helpompi ja nopeampi tapa, jossa esikatselukuva ilmestyi heti mikäli valittu tiedosto oli esikatseltavissa olevaa formaattia. Näin erilliselle sivulle ei tarvinnut avata kuvaa vaan sen näki heti pienessä ruudussa, jolloin siitä sai aikaan jo jonkinlaisen käsityksen. Tämä vahvisti myös käsitystä rakentaa erilliset ikkunat eri elementeille, jolloin myös esikatselulle voitaisiin tulevaisuudessa varata oma ikkunansa.

Useissa testatuissa järjestelmissä oli tekstitiedostojen muokkausmahdollisuus, jota ei kuitenkaan kehitettävässä järjestelmässä koettu tarpeelliseksi. Ainoat tekstitiedostot joita järjestelmään tultaisiin lähettämään, olisivat projektien kuvaustekstejä ja ne sisältäisivät yleensä kuvia ja muuta grafiikkaa, joten niitä olisi jokatapauksessa selkeämpi muokata ulkoisessa sovelluksessa. Tämä ominaisuus voisi toimia, mikäli järjestelmään olisi tarvittu jonkinlainen versionhallinta. Versionhallinta toteutetaan mahdollisesti kuitenkin vasta myöhemmässä vaiheessa, joten sekään ei tukenut päätöstä mahdollisen tiedostojen muokkaus ominaisuuden lisäämiseksi.

9.3 Ulkoasu

Miellyttävin ulkoasu oli WebInsta tiedonhallintajärjestelmässä. Muut ohjelmat olivat ulkoasun osalta jonkin verran hankalan ja keskeneräisen näköisiä. Ulkoasulla koettiin olevan todella merkittävä rooli ensivaikutuksen ja tässä tapauksessa myös toiminnallusuuden kannalta.

WebInsta on oman näköisensä, selkeä sekä miellyttävä katsoa ja käyttää. Kaikki graafiset elementit on tehty erikseen, ne ovat kuvaavia eivätkä aja käyttäjää harhaan yrittämällä matkia esimerkiksi Windowsin perusikoneita ja toimintamalleja. Käyttöliittymä on selkeä, vaikka tässäkin ohjelmassa on joitakin rakennettavalle järjestelmälle turhia ominaisuuksia ja elementtejä. Siitä huolimatta se on yksinkertainen ja nopea oppia. Omassa järjestelmässä päädyttiin käyttämään havainnollistavina ikoneina kunkin tiedostotyypin tiedostoikonia.

Ulkoasultaan ja ilmeeltään suunnitelmissa oli jotakin hieman WebInstan kaltaista. Tämä ohjelma vahvisti käsitystämme, sekä siitä saatiin hieman ideaa värimaailmaa ja yleisilmettä varten. Muut ohjelmat olivat todettu latteiksi ja liian teknisen näköisiksi. Järjestelmän haluttiin herättävän sen käyttäjissä positiivisia tunteita.

Yleiseen graafiseen ulkoasuun ei suurimmassa osassa testijärjestelmistä oltu suuresti panostettu, mutta käyttöliittymissä oli kuitenkin paljon eroja. Niin kuin aiemmin jo mainittiin, muiden käyttöjärjestelmien ulkoasujen matkimista pyrittiin välttämään koska ne loivat testijärjestelmiin osaltaan luotettavuutta mutta saattoivat ajaa myös käyttäjää harjaan. Koska suurin osa asiakkaista sekä Barabran työntekijöistä käyttää työtehtävissään Macintoshia, ei ulkoasua nähdä kannattavana rakentaa pelkästään Windowsin näköiseksi, kuten suurimmassa osassa valmiita ohjelmia oli toimittu. Pyrimme luomaan täysin uudenlaisen innovatiivisen käyttöliittymän uudenlaiseen ja yksinkertaiseen ulkoasuun puettuna.

10 Materiaalinhallintajärjestelmän kehittämisen vaiheet

Tiedonkeräämisen ja eri järjestelmien vertailun ja sen tulosten analysoimisen jälkeen aloitettiin opinnäytetyön käytönnön sovelluksen toteuttaminen. Määrittely- ja suunnitteluvaiheissa käytettiin vahvasti hyödyksi esimerkiksi muiden järjestelmien vertailussa ilmi tulleita asioita sekä yrityksessä järjestettyjen aivoriihien tuloksia. Tämä osio opinnäytetyöraportista sisältää järjestelmäkehityksen eri vaiheet, niissä hyödynnettyjen työkalujen esittelyt sekä osan sivutuotoksena tulleista kaavioista ja malleista.

10.1 Määrittely

Kun lähtökohdat ja uusmediatoimiston toimintamalli oli kartoitettu sekä muita ohjelmistoja vertailtu ja vertailun tuloksia analysoitu, ensimmäisenä vaiheena kehitysprojektiin valitussa

prosessimallissa oli määrittelyvaihe. Tässä vaiheessa toimeksiantajan kanssa yhteistyössä määriteltiin yrityksen tarpeet, järjestelmän ominaisuudet sekä järjestelmän eri käyttäjäryhmät tarkemmin. Määrittelyvaiheessa teoreettisena taustana toimii käytännössä koetut ongelmat ja haasteet joita yrityksen toiminnassa on todettu viime vuosien aikana, muut jo olemassa olevat järjestelmät sekä niiden benchmarking -tulokset, sekä aivoriihissä esiin tulleet ideat. Tässä vaiheessa järjestelmää lähdettiin kehittämään kehitysnimellä Magnet.

10.1.1 Vaatimusmäärittely

Yksi Rational Unified Process:n prosessimallin tärkeimmistä vaiheista on vaatimusten määrittely sekä niiden hallinta. Vaatimusmäärittelystä tekee tärkeän se, että koko ohjelmistokehitys perustuu siihen. Lisäksi asiakkaan, tässä tapauksessa Barabra Oy:n, näkökulmasta on tärkeää että kaikki ohjelmalle asetetut vaatimukset ovat kirjattu mahdollisimman tarkasti. (tSoft 2010.)

Opinnäytetyössä kehitettävälle järjestelmälle asetettiin tiettyjä vaatimuksia joita sen tulisi valmistuessaan täyttää. Vaatimukset voivat olla joko sen tekniseen toteutukseen, käyttäjiin tai käytettävissä oleviin resursseihin liittyviä. Seuraavassa on esiteltynä järjestelmälle asetettuja vaatimuksia.

Koska kehitettävän järjestelmän käyttäjäryhmiksi todettiin paljon niin osaamiseltaan kuin laitteistoltaankin erilaisia käyttäjiä, asetettiin järjestelmälle vaatimukseksi se että sen käyttäminen olisi helppoa ja tehokasta riippumatta siitä millaisella laitteistolla tai osaamisella käyttäjä järjestelmää käyttää. Järjestelmälle asetettiinkin vaatimukseksi nopea opittavuus ja ymmärrettävyys.

Teknisestä näkökulmasta järjestelmälle asetettiin vaatimukset liittyen toimintaan, sille annettuihin syötteisiin ja siitä lähteviin tulosteisiin. Oikeanlaisen käytön tuloksena järjestelmän tulisi toimia odotetulla tavalla, lähettää haluttu materiaali oikein sekä virhetilanteissa ilmoittaa käyttäjälle virheestä ja opastaa miten sen voisi korjata. Samat kriteerit pätevät myös järjestelmän luotettavuuteen ja sen vaatimuksiin. Järjestelmän tulisi myös toimia moitteettomasti ennalta määritellyillä www-selaimilla. Myöhemmässä vaiheessa on esitettynä erikseen selaimet joilla järjestelmän tuli toimia. Selainyhteensopivuuden lisäksi järjestelmän tulisi siis toimia virheettömästi käyttäjän näkökulmasta. Virheille, jotka eivät estä tai hankaloita järjestelmän käyttöä käyttäjän näkökulmasta, ei asetettu vaatimuksia.

Järjestelmän saatavuudelle asetettiin vaatimukseksi se että järjestelmä olisi saatavilla aina paitsi silloin kun saavutettavuus on palvelimen virheestä tai käyttäjän oman Internet-

yhteyden toimimattomuudesta johtuvaa. Kaikkiin muihin virheisiin Barabra Oy voisi itse reagoida ja korjata mahdolliset ongelmat.

Tässä vaiheessa määrittelyä tiedettiin että järjestelmän tehokkuuden mittareita on vaikea asettaa. Sen sijaan että sen tehokkuutta mitattaisiin ajassa, päätettiin käyttöönoton jälkeen seurata sitä kuinka vähän materiaalia enää siirrettään muilla keinoin, ja reagoida siihen tarpeen vaatiessa.

Barabra Oy:llä ei ollut projektiin varattuna rahallisia resursseja mutta yrityksen työntekijät olivat käytettävissä. Näin pystyttiin asettamaan resurssivaatimukseksi se että järjestelmä saadaan kehitetyksi omalla työvoimalla ilman rahallisia hankkeita. Henkilöstöresursseina käytettävinä oli Barabra Oy:n koko henkilökunta jolta pystyttiin pyytämään apua niin teknisissä kuin teoreettisissakin ongelmissa.

Edellä mainittujen vaatimusten lisäksi järjestelmälle asetettiin vaatimukseksi se että mahdollisten laitteistopäivitysten ja muutosten ajaksi järjestelmä on helposti siirrettävissä toiselle palvelimelle.

10.1.2 Erityistarpeiden määrittely

Heti projektin alkuvaiheessa alettiin tutkia ja tunnistaa järjestelmälle asetettuja erityisvaatimuksia. Ennen kehitysprojektia kohdeyrityksessä tehtiin tietoinen valinta siitä, että järjestelmä haluttiin tehdä alusta asti itse juuri omiin tarpeisiin sopivaksi, koska joitain OpenSource –vaihtoehtoja oli jo aiemmin kokeiltu ja koettu ne tarpeisiin sopimattomiksi. Tietoa kerättiin keskustelemalla halutuista ominaisuuksista kohdeyrityksen työntekijöiden kanssa, jolloin tunnistettiin järjestelmälle asetettavat erityistarpeet verrattuna muihin järjestelmiin.

Keskustelumuotona keskusteluissa käytettiin aivoriiheä, joka on yrityksen sisällä hyväksi todettu käytäntö. Aivoriihen avulla saadaan aikaan lukuisia ratkaisuehdotuksia ongelmiin, joista suurella todennäköisyydellä vähintäänkin pieni osa on erittäin käyttökelpoisia (Helsingin Yliopisto 2010). Keskustelujen pohjalta järjestelmälle asetettiin seuraavat erikoiskriteerit, jotka sen tulisi valmistuessaan täyttää. Näitä erikoistarpeita ja –ominaisuuksia olivat muun muassa:

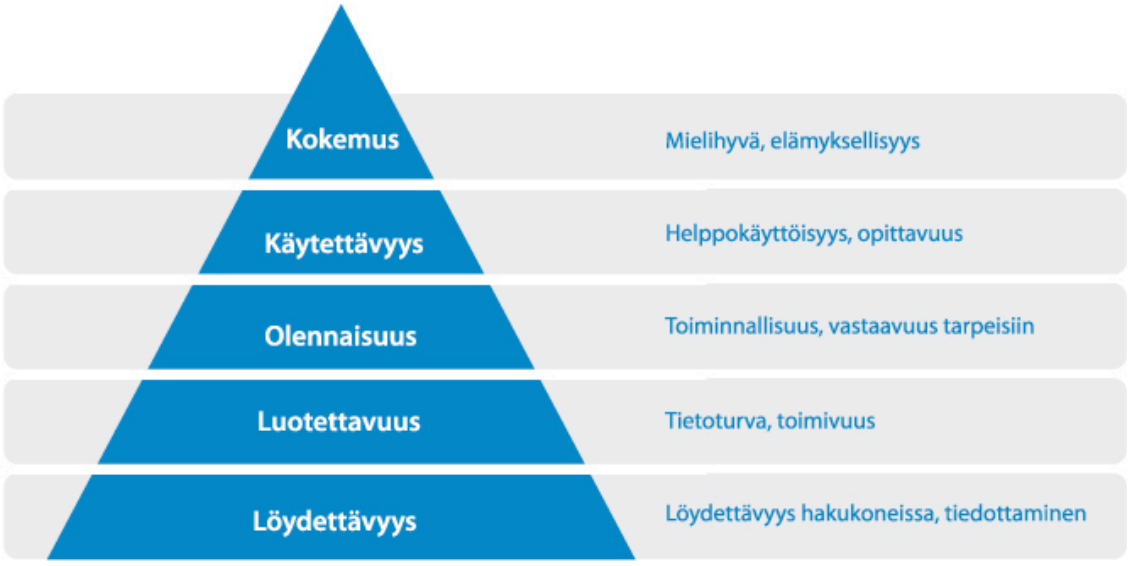
- Kohderyhmät ja niiden tarkka kartoittaminen
- Käyttäjälähtöisyys
- Selainriippumattomuus
- Ajasta ja paikasta riippumattomuus

Edes hyvin rakennettu järjestelmä ei ole hyödyllinen jos sille ei ole käyttäjiä jotka sitä tarvitsevat. Tietojärjestelmälle täytyy olla siis tarve, jonka määrittelee eri kohderyhmät. Vain asiakaskuntaa tunteva henkilö voi tunnistaa oleelliset tiedontarpeet. (Salmela 1999, 17.)

Koska järjestelmä ei tule pelkästään yrityksen sisäiseen käyttöön päätettiin ensin selvittää ja määritellä mahdolliset eri käyttäjäryhmät, joita järjestelmällä tulee olemaan. Koska ongelma materiaalinsiirrossa yrityksen sisältä asiakkaille ei ollut ainut tarve järjestelmän kehitykselle, yhdessä yrityksen johtohenkilöiden kanssa sovittiin, että järjestelmä suunnitellaan sopivaksi seuraaville kohderyhmille:

- asiakkaat (mainostoimistot, suorat asiakkuudet)
- yhteistyökumppanit (videotuotantoyritykset, mediatoimistot jne.)
- yrityksen työntekijät.

Käyttäjälähtöisyys on termi jolla halutaan kuvata tuotteen tai palvelun vastaavuutta käyttäjien tarpeisiin. Käytännössä käyttäjälähtöisyydellä tarkoitetaan siis ideaalista käyttökokemusta jonka saavuttaminen kokemuksen subjektiivisuudesta huolimatta tulisi olla jokaisen verkkopalvelun tavoite. (Valtionvarainministeriö 2008.)



Kuvio 1. Käyttäjälähtöisyyden eri osa-alueet (Valtionvarainministeriö 2008.)

Kohderyhmien määrittelyn jälkeen seuraavaksi erikoistarpeeksi kohdeyritys oli määritellyt, että järjestelmän tulisi palvella kaikkia eri kohderyhmiä tasavertaisesti. Käytettävyyden tulisi olla yhtä selkeää ja helppoa riippumatta siitä mihin kohderyhmään käyttäjä kuuluu. Suunnittelutyön pääteemaksi valittiinkin käyttäjälähtöisyys, ja sen pääpainopisteiksi käytettävyys ja olennaisuus.

Haasteena tässä kuitenkin oli että kaikki eri kohderyhmät omaavat huomattavasti erilaiset tietojenkäsittelytaidot ja heidän kokemuksensa erilaisista digitaalisista palveluista on hyvin vaihteleva (Jari Pelto-Piri, 2009). Kyseinen asia yritettiin ottaa huomioon jokaista eri toiminnallisuutta suunnitellessa.

Aivan kuin kohderyhmien osaaminenkin myös heidän laitteistonsa jolla he tulisivat järjestelmää käyttämään eroavat toisistaan huomattavasti. Kokemusten kautta on todettu että valtaosa mainosalalla toimivista yrityksistä hyödyntää Mac OS X pohjaista laitteistoa. Eroavaisuuksia laitteistossa kuitenkin havaitaan jo kohdeyrityksen itsensä sisällä. Osa työntekijöistä työskentelee täysin PC-pohjaisilla tietokoneilla, kun taas osa työskentelee työpaikallaan Mac OS X –pohjaisilla tietokoneilla. Heidän tulee kuitenkin päästä käsiksi järjestelmään myös kotoa käsin, jossa heillä on käytössään vain PC-tietokone.Tämän selvittyä lähdettiin kartoittamaan Internet-selaimia, joissa järjestelmän tulisi toimia.

Taulukko 3. Eri internet-selainten käyttäjämäärät (W3Schools 2009.)

2009	IE8	IE7	IE6	Firefox	Chrome	Safari	Opera
August	10.6%	15.1%	13.6%	47.4%	7.0%	3.3%	2.1%
July	9.1%	15.9%	14.4%	47.9%	6.5%	3.3%	2.1%
June	7.1%	18.7%	14.9%	47.3%	6.0%	3.1%	2.1%
May	5.2%	21.3%	14.5%	47.7%	5.5%	3.0%	2.2%
April	3.5%	23.2%	15.4%	47.1%	4.9%	3.0%	2.2%
March	1.4%	24.9%	17.0%	46.5%	4.2%	3.1%	2.3%
February	0.8%	25.4%	17.4%	46.4%	4.0%	3.0%	2.2%
January	0.6%	25.7%	18.5%	45.5%	3.9%	3.0%	2.3%

Eri selainten käyttäjämäärien selvittämisen jälkeen päädyttiin seuraavaan johtopäätökseen: jotta järjestelmä toimisi moitteettomasti kaikilla sille asetetuilla kohderyhmillä, sen tulisi toimia täysin moitteettomasti tietyillä selaimilla. Yllä esitetyn taulukon (Taulukko 1) mukaan pystyttiin valitsemaan tietyt selaimet joilla järjestelmä testattaisiin, jolloin se toimisi lähes kaikilla selaimilla, koska muut selaimet käyttävät samaa HTML –tulostusmoottoria kuin valitut selaimet. Järjestelmä testattiin siis seuraavilla selaimilla, joiden yhteinen kattavuus kaikista käytetyistä selaimista on noin 80%.

- Mozilla Firefox
- Google Chrome
- Internet Explorer 7 ja 8

Järjestelmälle hyvin tärkeää on myös se että sen sisältämä tieto on jaettavissa verkon välityksellä etäisyydestä, aikaerosta ja henkilökunnan työajoista rippuumatta. Tiedon on oltava siis saatavilla juuri silloin kun käyttäjä sitä tarvitsee. (Juha Salmela, 1999, 16.) Jo varhaisessa vaiheessa suunnitteluprosessia voidaan olla varmoja että järjestelmästä tulee www-

selaimessa toimiva, joka mahdollistaa sen että se on saatavilla ympäristötekijöistä riippumatta.

10.1.3 Järjestelmän käyttäjäryhmät

Järjestelmää tulevat käyttämään lukuisat eri asiakkaat, joten järjestelmässä tulee olla erikseen käyttäjienhallinta. Järjestelmää tulee käyttämään kokonaisuudessaan kolme eri pääkäyttäjäryhmää: asiakkaat, yhteistyökumppanit sekä Barabran oma henkilöstö. Esimerkiksi tietoturvasyistä asiakkaat eivät saa nähdä toistensa kansioita ja dokumentteja. Asiakkailla ei myöskään saa olla pääsyä muihin kuin heille jaettuihin materiaaleihin sekä heidän itsensä lataamiin materiaaleihin.

Barabran sisällä kuitenkin on tarvetta jakaa käyttäjät kahteen eri ryhmään. Toisella ryhmällä on oikeudet lisätä uusia käyttäjiä sekä pääsy kaikkiin kansioihin ja tiedostoihin, kun taas toisella ryhmällä on oikeus päästä käsiksi kaikkiin järjestelmän materiaaleihin, mutta ei ole oikeutta lisätä tai poistaa uusia käyttäjiä.

Tämän lisäksi, järjestelmästä on mahdollista lähettää tiedostoja ladattavaksi henkilöille, jotka eivät välttämättä edes omista tunnuksia järjestelmään. Seuraavana on esiteltynä erilaisia käyttäjätunnutstyyppejä, joita tässä vaiheessa määrittelyä järjestelmälle asetettiin. Eri käyttäjätunnustyypeiksi määriteltiin:

- täydet oikeudet omaava käyttäjä
- peruskäyttäjä
- käyttäjä jolla on oikeuedet vain omaan kansioon
- ulkoinen käyttäjä.

10.1.3.1 Täydet oikeudet omaava käyttäjä

Täydet oikeudet omaavia käyttäjiä tulee olemaan Barabran johtohenkilöt, joilla on oikeus lisätä asiakkaita järjestelmään sekä luoda uusia tunnuksia. Näillä käyttäjillä on oikeus poistaa, lisätä sekä muokata käyttäjiä. Täydet oikeudet omaavalla käyttäjällä on myös mahdollisuus nähdä kaikkien asiakkaiden ja käyttäjien kansiot ja tiedostot, sekä muokata ja poistaa mitä vain materiaalia. Täydet oikeudet omaava käyttäjä näkee myös erillisestä näkymästä kaikki uusimmat ladatut tiedostot sekä kansiot.

Barabralla tulee olla kaikki oikeudet järjestelmän käytössä, jotta se voi hallinnoida järjestelmän käyttäjiä, lisätä uusia asiakkaita, sekä poistaa ja muokata käyttäjiä, sekä tiedostoja. Barabran sisällä voidaan luoda työntekijöille uusia tunnuksia sen mukaan mitä kukin tarvitsee

omissa projekteissaan. Johtohenkilöstö voi käyttää täydet oikeudet omaavia käyttäjiä ja luoda sitä mukaan tunnuksia työntekijöille kun on tarvetta.

Tämän tyyppinen käyttäjätunnus luokitellaan järjestelmän ylläpitäjäksi. Tunnuksilla voi käyttää järjestelmän kaikkia toimintoja, jotka ovat:

- tiedostojen ja kansioiden lisääminen järjestelmään
- tiedostojen ja kansioiden poistaminen järjestelmästä
- tiedostojen selaaminen
- tiedostojen esikatselu
- tiedostojen lähettäminen käyttäjille
- näkymä järjestelmän uusista tiedostoista
- näkymä järjestelmän uusista kansioista
- mahdollisuus lähettää ja vastaanottaa viestejä.

10.1.3.2 Peruskäyttäjä

Rajoitetuilla oikeuksilla olevia tunnuksia käyttävät Barabran työntekijät. Näillä käyttäjillä on muilta osin samat oikeudet nähdä kansiot ja tiedostot kuin täydet oikeudet omaavilla käyttäjillä, mutta eroavaisuutena rajoitetut oikeudet omaava käyttäjä ei voi lisätä uusia käyttäjiä tai muokata toisten käyttäjien tietoja.

Koska asiakkailla on mahdollisuus nähdä vain omat tiedostonsa, ei asiakkaille anneta peruskäyttäjän oikeuksia. Barabran henkilöstöllä on kuitenkin tarve nähdä monen eri asiakkaan materiaalit.

Peruskäyttäjälle sallittuja toiminnallisuuksia ovat:

- tiedostojen ja kansioiden lisääminen järjestelmään
- tiedostojen ja kansioiden poistaminen järjestelmästä
- tiedostojen selaaminen
- tiedostojen esikatselu
- tiedostojen lähettäminen käyttäjille
- näkymä järjestelmän uusista tiedostoista
- näkymä järjestelmän uusista kansioista
- mahdollisuus lähettää ja vastaanottaa viestejä.

10.1.3.3 Oikeudet vain omaan kansioon

Nämä oikeudet annetaan asiakkaille, joille halutaan oikeus luoda oma kansio, jonne materiaaleja voi lisätä, poistaa ja muokata. Asiakkailla on kuitenkin oikeus vain omaan kansioon, eikä toisten asiakkaiden tiedostoihin ja kansioihin ole oikeuksia.

Näitä oikeuksia ei käytetä Barabran työntekijöillä siitä syystä, että usein moni eri työntekijä tekee töitä saman asiakkaan projekteissa. Heillä tulee olla oikeudet nähdä kaikki asiakkaat ja näiden materiaalit. Esimerkiksi sairastapauksissa jonkun toisen työntekijän on otettava vastuulleen sairaslomalla olevan työtehtävät, ja näin ollen on hyvä että jokaisella Barabran työntekijällä on oikeudet jokaiseen kansioon, jotta materiaaleihin pääsee helposti ja vaivattomasti käsiksi omilla tunnuksilla, ilman että täydet oikeudet omaavan käyttäjän pitäisi muuttaa tunnusten oikeuksia tai tehdä uusia väliaikaisia tunnuksia aina poikkeustapauksissa.

Tämäntyyppisellä käyttäjätunnuksella tulee olla siis seuraavat toiminnallisuudet:

- tiedostojen ja kansioiden lisääminen omaan kotihakemistoon
- tiedostojen ja kandisoiden poistaminen omasta kotihakemistosta
- oman kotihakemiston selaaminen
- oman kotihakemiston tiedostojen esikatselu
- tiedostojen lähettäminen käyttäjille
- näkymä oman kotihakemiston uusista tiedostoista
- näkymä oman kotihakemiston uusista kansioista
- mahdollisuus lähettää ja vastaanottaa viestejä.

Myös asiakkaan näkökulmasta materiaalinhallinta helpottuu Barabran kanssa, koska materiaalit löytyvät aina yhdestä paikasta eikä niitä tarvitse erikseen tallentaa omalle koneelle. Usein voi käydä niin että asiakas on tallentanut tiedostot omalla koneella eri paikkoihin ja saattaa jossakin tilanteessa palata katsomaan väärää aineistoa, mikä ei ole ajantasalla. Uusi materiaalinhallintajärjestelmä auttaa asiakasta löytämään helposti uusimman tiedoston. Asiakas voi katsoa sähköpostista uusimman viestin mikä on tullut Barabran materiaalinhallintajärjestelmästä ja löytää linkin takaa uusimman tiedoston ongelmitta.

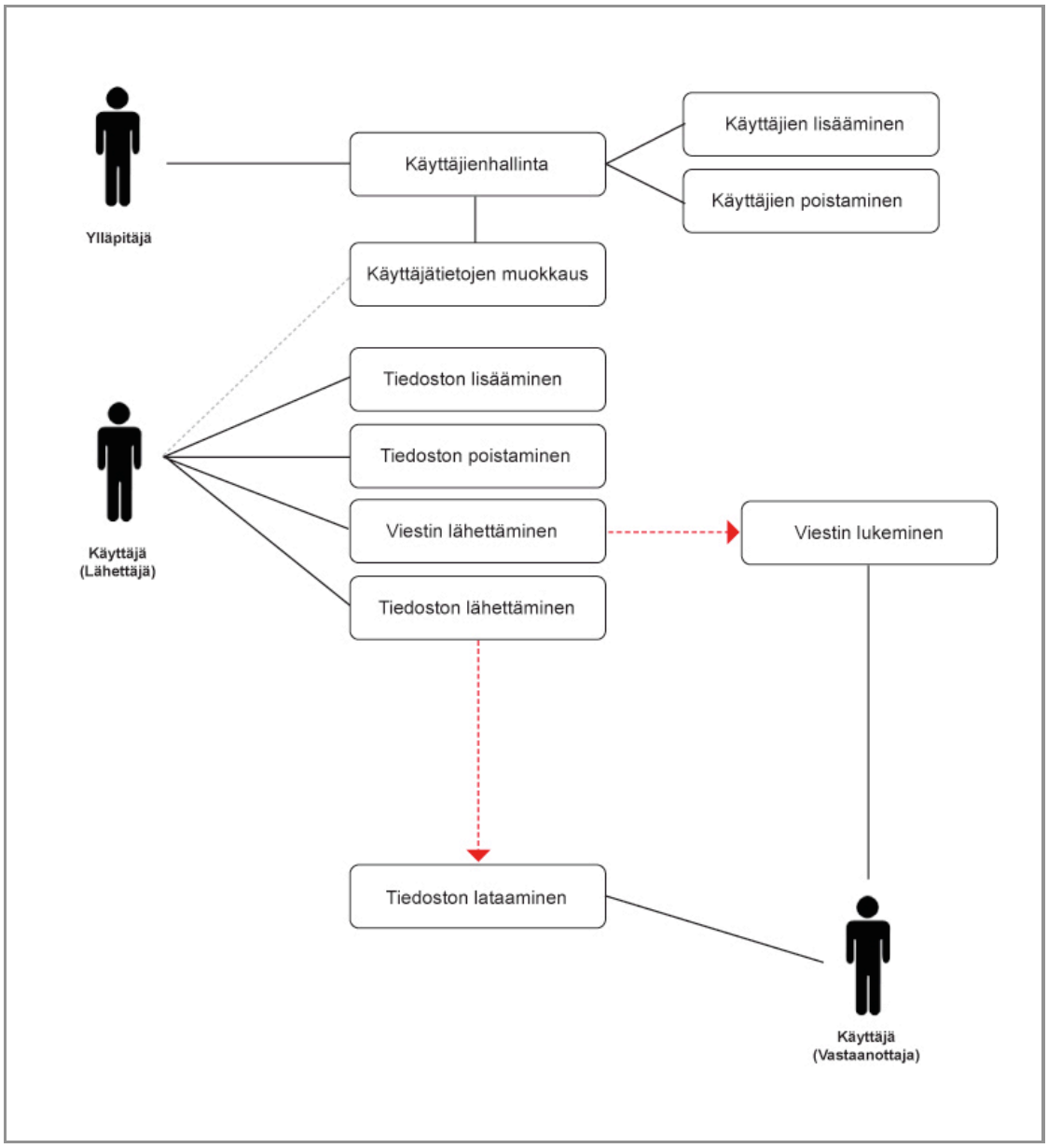
10.1.3.4 Ulkoiset käyttäjät

Materiaalinhallintajärjestelmää voi osaltaan hyödyntää myös käyttäjät, joilla ei ole järjestelmään tunnuksia. Näitä käyttäjiä kutsutaan termillä “ulkoinen käyttäjä”. Nämä käyttäjät käyttävät järjestelmää ainoastaan materiaalin noutamiseen järjestelmästä.
Ulkoisia käyttäjiksi luokitellaan muun muassa alihankkijat, painotalot sekä mahdolliset freelancertyöntekijät joille jaetaan materiaalia. Näille käyttäjille ei kuitenkaan ole tarvetta an-

taa oikeuksia järjestelmään, koska pääsääntöisesti alihankkijoiden, painotalojen sekä freelacertyöntekijöiden kanssa materiaalia jaetaan vain hyvin tapauskohtaisesti.

10.1.4 Järjestelmän toiminnallisuudet

Järjestelmän ominaisuudet määriteltiin pääasiassa tutkimalla olemassa olevia vastaavia järjestelmiä sekä aivoriihissä syntyneiden ideoiden pohjalta. Valmiiden järjestelmien vahvuuksia päätettiin hyödyntää sekä hiiden heikkouksia päätettiin pyrkiä välttämään. Niiden perusteella kehitettiin juuri kohdeyritykselle sopiva järjestelmä. Järjestelmän testausvaiheessa tullaan olemaan yhteydessä asiakkaisiin, sekä lisäksi pohditaan Barabran sisällä nykyisien ominaisuuksien tarpeellisuutta, hyödyllisyyttä ja käytettävyyttä. Lisäksi keskustellaan mahdollisista parannusehdotuksista ja uusista ominaisuuksista mitä järjestelmään tulisi saada.



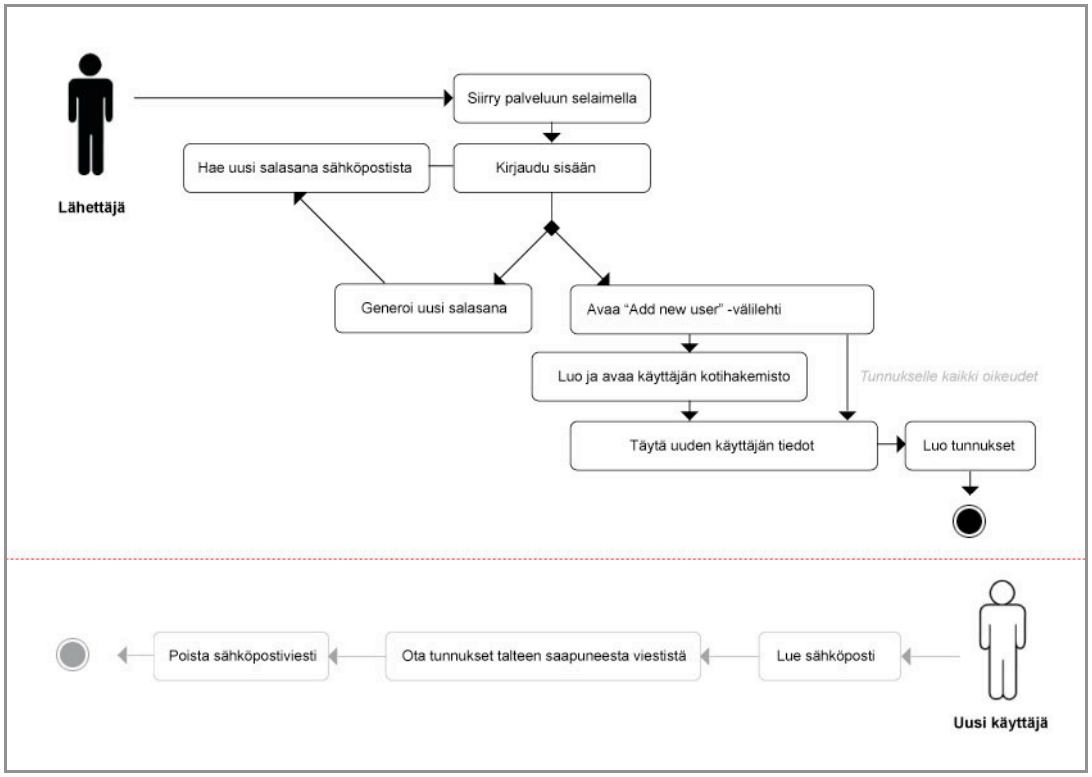
Kaavio 1. Järjestelmän käyttötapaukset kuvattuna käyttötapauskaaviona

Edellä olevassa kaaviossa on esiteltynä järjestelmän eri käyttötapaukset kootusti yhdessä käyttötapauskaaviossa. Käyttötapauskaavio sisältää käyttötapaukset järjestelmän ominaisuuksista, jotka päätettiin sisällyttää heti järjestelmän ensimmäiseen versioon. Seuraavana on esiteltynä kukin käyttötapaus tarkemmin toimintokaavioina. Kaikki nämä toiminnot koettiin välttämättömiksi järjestelmän toimivuuden ja tarpeellisuuden kannalta.

10.1.4.1 Käyttäjätunnuksiin liittyvät toiminallisuudet

Käyttäjien lisääminen on yksi järjestelmän ydinominaisuuksista. Käyttäjän lisääminen tarkoittaa samalla myös uusien asiakkaiden lisäämistä. Tätä ominaisuutta ei anneta Barabran ulkopuolelle, jolloin vältetään järjestelmän väärinkäyttö ja tietoturva säilytetään asiakkaiden sekä omien materiaalien osalta mahdollisimman pitkälle.

Käyttäjiä voi lisätä ainoastaan täydet oikeudet omaava käyttäjä. Käyttäjien lisäämisen yhteydessä tulee olla mahdollista myös lisätä ja muokata käyttäjän tietoja ja oikeuksia. Uuden käyttäjän luomiseen tarvitaan käyttäjän haluama käyttäjätunnus sekä sähköpostiosoite. Kun uusi käyttäjä luodaan, syötetään järjestelmään asiakkaan haluama käyttäjätunnus sekä sähköpostiosoite Tämän lisäksi tunnuksia luodessa määritellään asiakkaalle järjestelmän käyttöoikeudet. Kun uusi käyttäjä on onnistuneesti lisätty, järjestelmä lähettää asiakkaalle tämän ilmoittamaan sähköpostiosoitteeseen tunnukset, joilla asiakas voi kirjautua järjestelmään ja aloittaa sen käyttämisen.

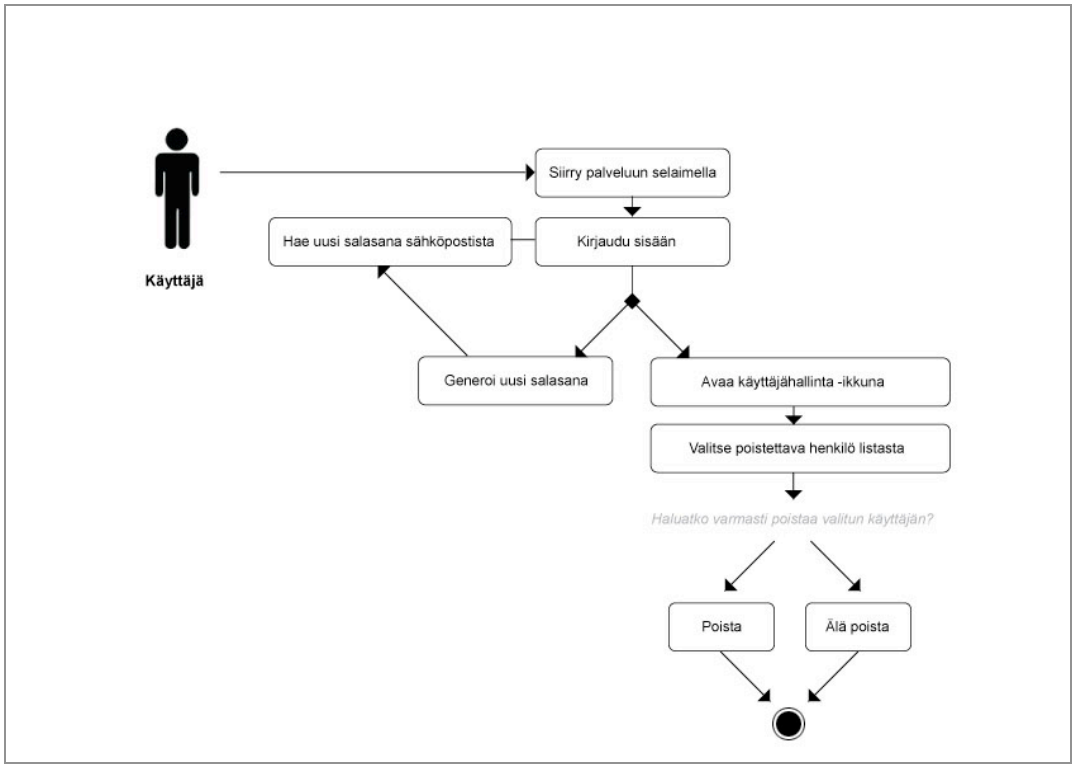


Kaavio 2. Käyttäjän lisääminen esitetty toimintokaaviona

Uuden käyttäjän luomisen yhteydessä tietojärjestelmään tallentuu uuden käyttäjän profiili, jonka tietoja voi käyttäjä itse halutessaan muuttaa. Kukin käyttäjä voi muo-kata profiilissa olevia tietoja milloin tahansa. Profiilissa pakolliset kentät ovat käyttäjätunnus sekä säh-köposti.

Kaikkea muuta paitsi käyttäjätunnusta voi muuttaa jälkikäteen. Myös salasanaa voi muuttaa, mikäli automaattisesti generoitu salasana ei käyttäjää jostain syystä miellytä. Salasanan unohtumiseen on pienempi riski, mikäli käyttäjä voi määritellä sen itse. Lisäksi se lisää käyt-tömukavuutta. Salasanat ovat tallennettuna tietokantaan MD5-tiivisteinä, joten niiden palaut-taminen ei ole mahdollista. Unohtaessaan salasanan voi käyttäjä kuitenkin generoida uuden salasanan, joka lähetetään hänen profiilissa olevaan sähköpostiin. Ainoastaan kaikki oikeudet omaava käyttäjä voi muokata myös muita kun oman käyttäjätilinsä tietoja.

Käyttäjiä voi poistaa ainoastaan täydet oikeudet omaava käyttäjä. Käyttäjiä kuitenkaan har-voin poistetaan, koska yleensä jos asiakkuussuhde on niin vakaalla pohjalla että asiakkaalle on toimitettu pääsy järjestelmään, voidaan olettaa että yhteistyö jatkuu useamman projektin yhteydessä. Jos tilanne vaatii, on profiilien poisto kuitenkin mahdollista. Asiakkaat eivät itse voi poistaa omaa profiiliaan järjestelmästä. Käyttäjiä voi poistaa vaan Barabra, ettei mahdol-lisia vahinkoja ilmene. Barabra voi myös pii-lottaa tunnuksen vain väliaikaisesti, mikäli on oletettavaa että tunnuksia tarvitaan vielä jatkossa.



Kaavio 3. Käyttäjän poistaminen esitetty toimintokaaviona

10.1.4.2 Sisällön lisääminen

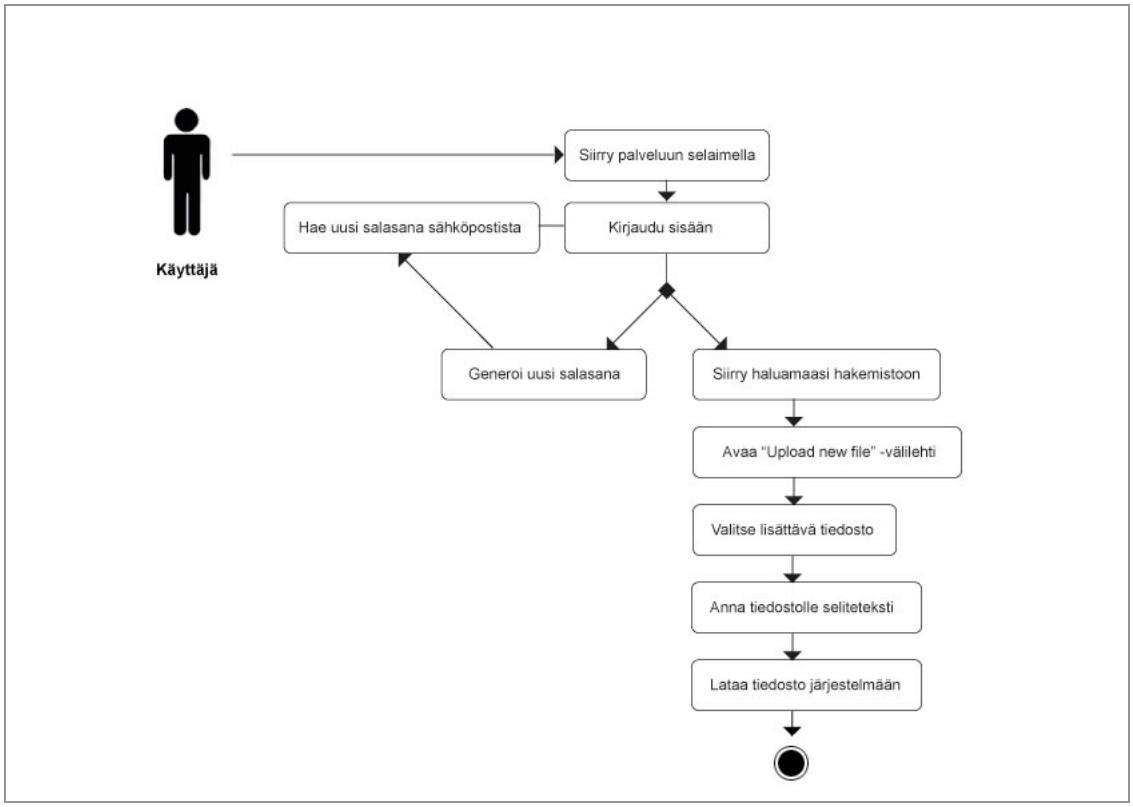
Käyttäjät voivat lisätä järjestelmään kansioita ja nimetä ne haluamikseen. Riippuen käyttäjän käyttöoikeuksista käyttäjä voi luoda uusia kansioita yhteen tai useampaan eri paikkaan.

Asiakkaat voivat luoda ja poistaa alikansioita omassa kansiossaan, mutta heillä ei ole pääsyä toisten asiakkaiden kansioihin, eivätkä asiakkaat näe toistensa kansioita. Myöskään muiden asiakkaiden kansiot eivät näy ”Uusimmat kansiot” –listassa, joka tulee näkymään päänäkymässä. Kansioille voi antaa haluamansa nimen, ja nimeä voi myös muuttaa tai vaihtaa jälkikäteen.

Kansioiden lisäksi järjestelmään voi lisätä mitä tahansa tiedostoja paitsi käyttöjärjestelmässä ajettaviksi tarkoitettuja tiedostoformaatteja. Tämä kattaa esimerkiksi .exe –päätteiset tiedostot. Tähän ratkaisuun päädyttiin tietoturvasyistä. Riippuen käyttöoikeuksista tiedostoja voi lisätä kuitenkin vain niihin kansioihin mihin käyttäjätunnuksella on oikeudet.

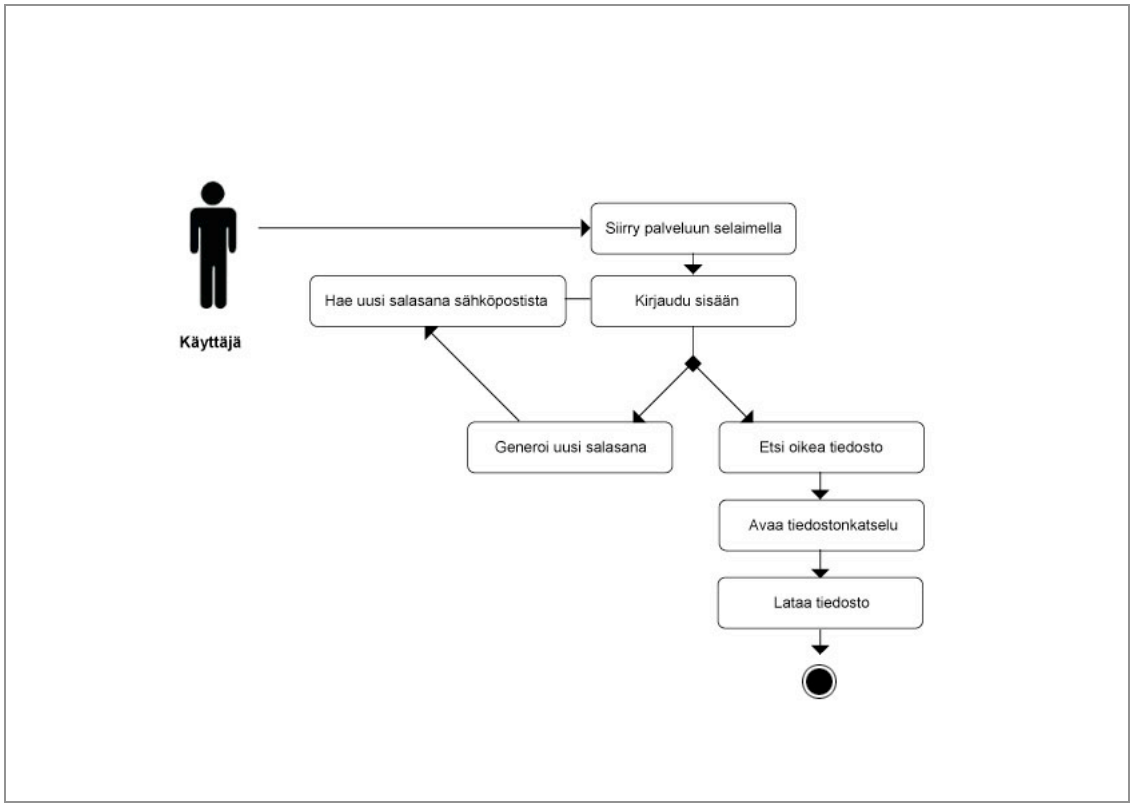
Tiedostojen lisäämisen yhteydessä tiedostoon voi liittää kuvaustekstin, jossa voi kertoa tarkemmin mitä tiedosto sisältää. Usein tiedoston nimellä yritetään kuvata mahdollisimman tarkasti, mitä tiedosto pitää sisällään, mutta kuvaustekstin avulla voidaan välttää ylipitkiä tiedostonnimiä sekä kuvata vapain sanoin mitä tiedosto pitää sisällään.

Kuvaustekstin avulla voidaan myös osittain selkeyttää mikä versio kyseinen tiedosto on. Usein kun materiaalia jaetaan kahteen suuntaan asiakkaan ja Barabran välillä, samasta tiedostosta luodaan monta eri versiota, jolloin yleensä tiedostot tulee nimetä erilailla, niin että niistä selviää mikä versio on kyseessä ja mikä on sillä hetkellä uusin versio kyseisestä materiaalista. Yleensä versionhallinta toteutetaan muokkaamalla aina tiedoston nimeä lisäämällä siihen perään numero, joka kasvaa aina, kun uusi versio tiedostosta luodaan. Tähän oli suunnitelmissa jonkinlainen versionhallinta, niin että järjestelmä olisi automaattisesti esittänyt tiedostojen osalta uusimman tiedoston. Tästä ominaisuudesta kuitenkin päätettiin ensimmäisessä versiossa luopua projektin yleisen laajuuden vuoksi.



Kaavio 4. Tiedoston tai kansion poistaminen esitetty toimintokaaviona

Kukin käyttäjä voi ladata tiedostoja kansioista, joihin hänellä on käyttöoikeudet. Lataamisen yhdeydessä tulee ilmoitus, mitä tiedostoa on lataamassa, ennen kuin latauksen voi aloittaa. Tämä ehkäisee sen, että käyttäjä lataisi vahingossa esimerkiksi väärän tiedoston.

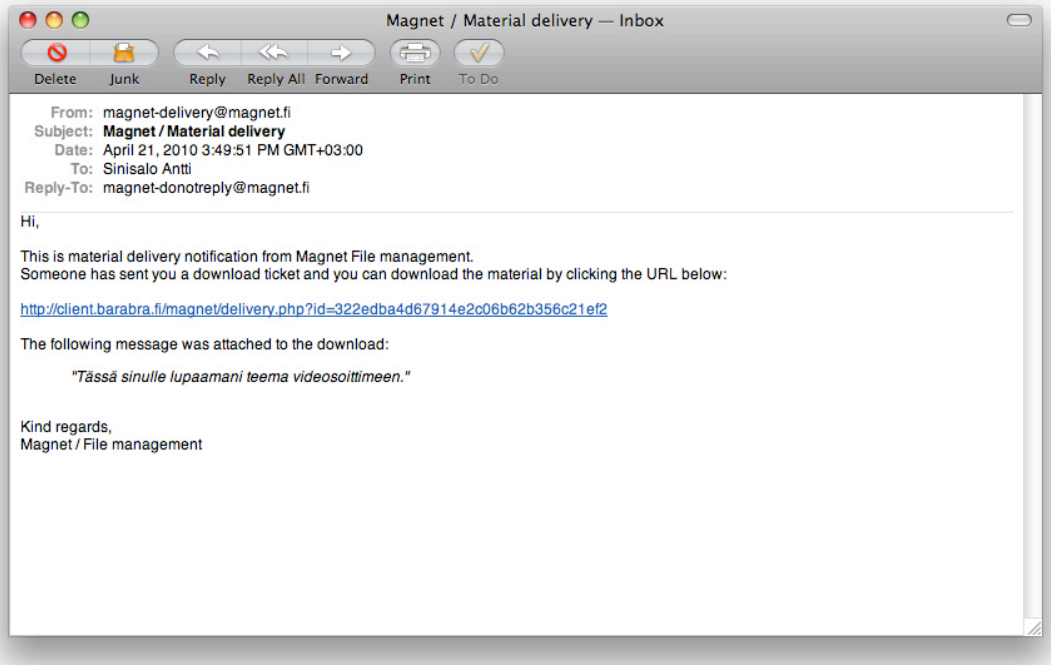


Kaavio 5. Tiedoston lataaminen esitetty toimintokaaviona

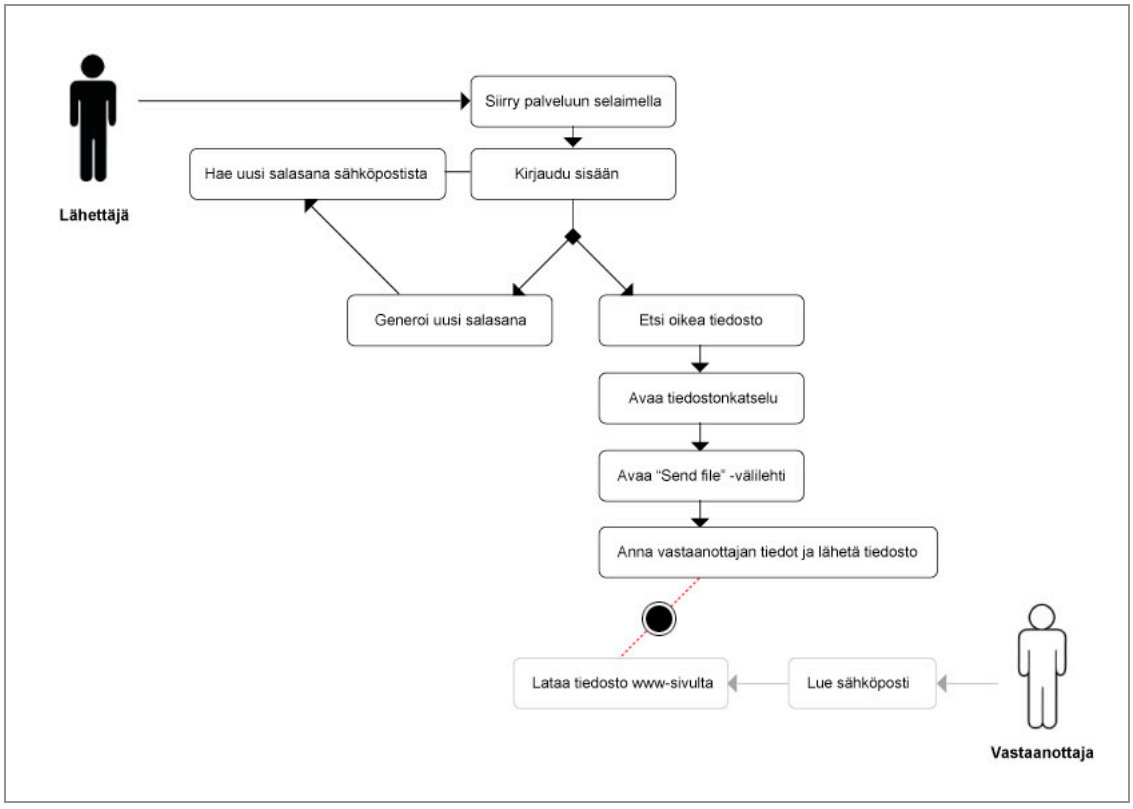
Tiedostoja voi selata ja tietyntyyppisiä mediatiedostoja esikatsella niissä kansioissa, joihin oikeudet riittävät. Tiedostojen esikatselussa tiedostosta näkee tiedostokoon, tiedostotyypin ja nimen.

10.1.4.3 Sisällön lähettäminen

Uusista tiedostoista voi ilmoittaa vastaanottajalle osapuolelle sähköpostilla suoraan järjestelmän sisältä ilman että tarvitsee avata esimerkiksi omaa sähköpostiohjelmaa. Järjestelmä lähettää sähköpostin linkillä varustettuna haluttuun sähköpostiosoitteeseen. Linkin takaa vastaanottaja voi käydä lataamassa kyseisen tiedoston omalle koneelleen.



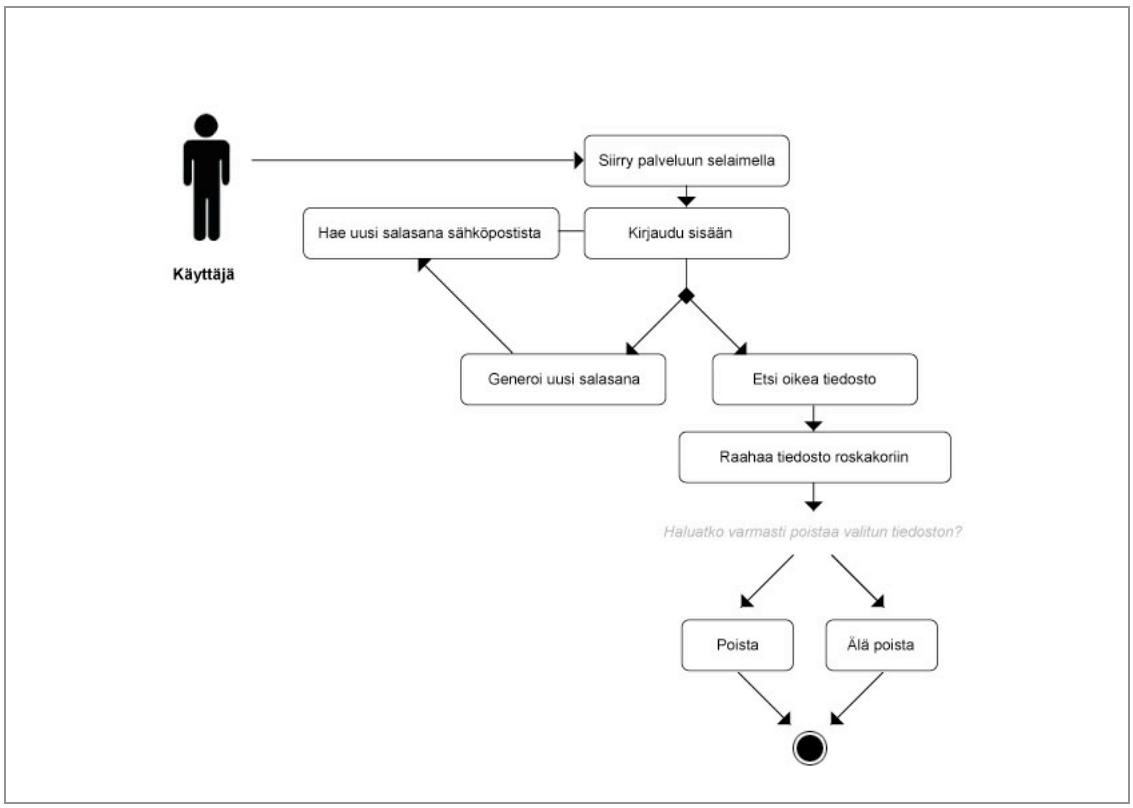
Kuva 2. Kuva järjestelmästä lähetetystä latauslinkistä



Kaavio 6. Tiedoston latauslinkin lähettäminen esitetty toimintokaaviona

10.1.4.4 Sisällön poistaminen

Kullakin käyttäjällä on mahdollisuus poistaa tiedostoja ja kansioita, joihin omaa oikeudet. Käyttäjillä on oikeus poistaa kaikki kansiot, jotka käyttäjä on lisännyt järjestelmään.



Kaavio 7. Tiedoston tai kansion poistaminen esitetty toimintokaaviona

10.1.4.5 Muu toiminnallisuus

Selkeiden perustoimintojen lisäksi järjestelmään määriteltiin monia muitakin ominaisuuksia joilla sen käyttöä tultaisiin helpottamaan. Esimerkiksi ladatut tiedostot ilmestyvät listana järjestelmän päänäkymään, josta näkee uusimmat tiedostot. Uusimman ladatun tiedoston näkee heti päänäkymästä, joten kansioita ei tarvitse käydä läpi etsiäkseen uusia tiedostoja, vaan uusimman ladatun tiedoston nimi on heti järjestelmään kirjauduttua näkyvillä.

Viimeisimmät tiedostot näkee ainoastaan käyttäjä, jolla on niihin oikeus. Tietoturva- sekä salassapitosyistä ei ole hyväksyttävää näyttää kaikkien järjestelmässä sijaitsevien materiaalien virtaa jokaisen käyttäjän kesken. Kullekkin käyttäjälle näytetään vain ne tiedostot ja kansiot, mihin oikeudet riittävät.

10.2 Suunnittelu

Määrittelyvaiheen jälkeen siirryttiin projektin suunnitteluvaiheeseen, jossa järjestelmään suunniteltiin tarkasti järjestelmälle aiemmin määritellyt toiminnallisuudet, ominaisuudet, käyttöliittymät, tietokannat ja niiden rakenteet sekä käytettävät teknologiat, työkalut, tekniset ratkaisut ja käyttöliittymä. Tässä vaiheessa prosessia yritettiin hyödyntää mahdollisia ohjelmistoteollisuuden käyttämiä yleisesti hyväksi todettuja suunnittelumalleja ja käytäntöjä.

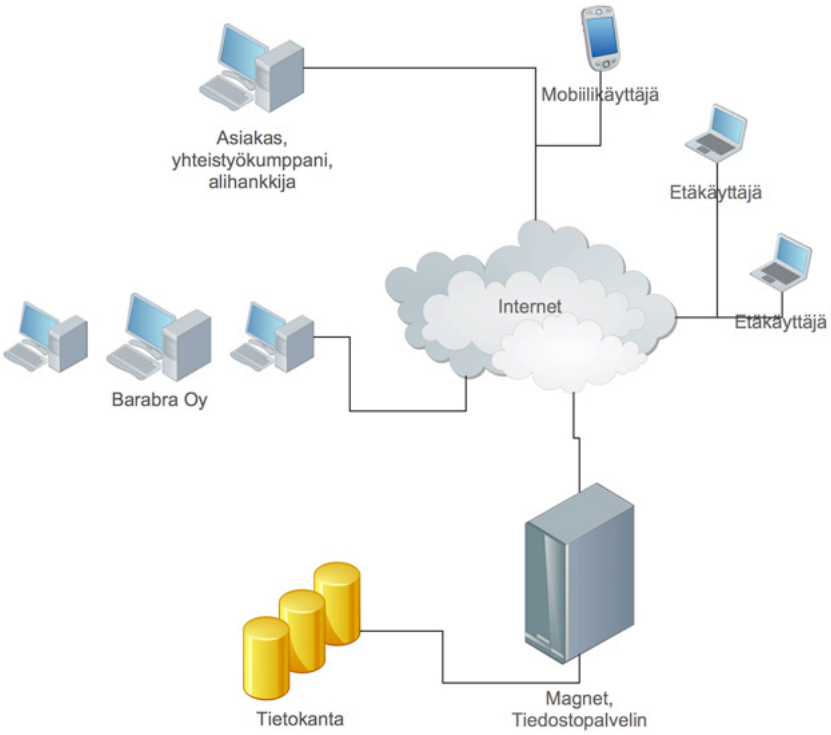
10.2.1 Toteutukseen käytettävät tekniikat

Kehitettävän materiaalinhallintajärjestelmän tulee palvella niin itse kohdeyritystä, yhteistyökumppaneita kuin myös sen asiakasyrityksiä. Tästä syystä järjestelmä päätettiin toteuttaa täysin selainpohjaisena, jotta Barabran työntekijät sekä asiakasyrityksen vastuuhenkilöt pääsevät sinne käsiksi fyysisestä sijainnistaan huolimatta. Selainpohjaisen järjestelmän toteuttamisesta oltiin yrityksen sisällä yksimielisiä.

Koska järjestelmä on täysin www-pohjainen, looginen valinta tietokannaksi oli MySQL. Itse ohjelmakoodi on pääasiassa PHP:ta, ja käyttöliittymä toteutettiin käyttäen HTML:ää. Näiden lisäksi voi olla mahdollista järjestelmään toteutettiin mittavia osia esimerkiksi käyttäen AJAX –teknologiaa. Ennen käyttöliittymän siirtämistä HTML –muotoon, se suunniteltiin käyttäen Adobe Photoshoppia.

10.2.2 Rakenne

Materiaalinhallintajärjestelmä koostuu tietokantapalvelimesta, käyttöliittymästä (websivu) sekä verkkopalvelimesta. Järjestelmän rakenne on rakennettu mahdollisimman yksinkertaiseksi ja loogiseksi järjestelmän rakentamisen ja jatkokehityksen helpottamiseksi. Oheisessa kuvassa (Kuva 5) on esitelty yleisnäkymä järjestelmän infrastruktuurista.



Kaavio 8. Yleisnäkymä järjestelmän infrastruktuurista

Materiaalinhallintajärjestelmässä käytetään MySQL-tietokantapalvelinta, joka on yleisesti yksi käytetyimmistä tietokantapalvelimista ja soveltuu hyvin pienempiin sekä massiivisempiinkin järjestelmiin. MySQL-tietokantapalvelimen valinta projektiin oli helppo, koska se on ennestään tuttu sekä valmiiksi käytössä Barabra:ssa.

Tiedostopalvelimena käytetään Barabran FTP-palvelinta, jota varsinaisesti käytetään materiaalien arkistointiin ja jakamiseen asiakkeille sekä yrityksen työntekijöille. Materiaalinhallintajärjestelmä käyttää FTP-palvelinta tiedostojen ylläpitämiseen, ja tiedostojen käsittely palvelimella käyttäen ftp:tä loppuu järjestelmän käyttöönoton yhteydessä.

Järjestelmä on sijoitettu palvelimelle omaan kansioonsa, jonka alakansioihin on sijoitettu järjestelmän eri osia sekä järjestelmän kautta tallennetut tiedostot. Kuvat, scriptitiedostot, tyylitiedostot ja muut erilaiset tiedostotyypit on sijoitettu palvelimelle eri kansioihin, jotta järjestelmä olisi kehittäjälle helpommin lähestyttävä sekä helpompi ja nopeampi tarkastella.

Tiedostot tallennetaan fyysisesti FTP-palvelimelle ja MySQL-tietokantaan tallennettavan tiedon perusteella fyysisistä tiedostoista rakennetaan järjestelmään visuaalinen tiedostorakenne. Käyttäjän ei tarvitse ottaa yhteyttä FTP-palvelimeen vaan järjestelmä luo yhteydet käyttäjän puolesta, tuo tiedostot tarkasteltavaksi sekä tallentaa halutut tiedostot palvelimelle.

10.2.3 Ohjelmamoduulit

Itse järjestelmä koostuu lukuisista eri ohjelmamoduuleista. Järjestelmän ohjelmakoodit on jaettu loogisesti ja järjestelmällisesti eri tiedostoihin, josta niitä on helppo ja nopea tarkastella. Myös itse ohjelmalogiikka nopeutuu.

Järjestelmän käyttöliittymä on pääosin ohjelmoitu AJAX –scriptikielellä ja rakenne on HTML -merkkauskielellä rakennettu. Rakennetiedostoon liitetään ajon yhteydessä lukuisa määrä tiedostoliitteitä, joita rakennetiedosto käyttää myöhemmin tarpeen tullen. Rakennetiedosto sisältää perustiedot standardeista, metadatasta, sekä järjestelmän ylläpitäjistä. Rakennetiedostoon liitetään erillisinä tiedostoina muun muassa järjestelmän tyylimäärittelyt, ohjelmakirjastot sekä tietokantayhteydet. Tämä nopeuttaa ohjelmakoodin läpikäymistä niin järjestelmän puolelta kuin kehittäjänkin näkökulmasta.

10.2.4 Käyttöliittymä

Määrittelyn ja tietokannan suunnittelun jälkeen tiedossa oli järjestelmässä tarvittavat ominaisuudet, ja näin ollen pystyttiin aloittamaan käyttöliittymän hahmottaminen. Käyttöliittymän suunnittelemiseen käytettiin Adobe Systemin tuoteperheen eri ohjelmistoja, koska ne olivat jo ennestään kohdeyrityksen käytettävissä.

Käyttöliittymän perusnäkymä ja graafiset elementit suunniteltiin Adoben Photoshop CS4 -ohjelmistolla. Adobe Photoshop oli Barabrassa käytössä jo valmiiksi, joten käytännössä mitään muuta vastaavat ominaisuudet sisältävää ohjelmaa ei voitu ilman lisäkustannuksia ottaa käyttöön. Photoshop oli järjestelmän kehittäjillekin jo ennestään tuttu, joten sen käyttäminen projektissa tuntui luontevalta vaihtoehdolta. Yrityksen sisällä käytyjen keskusteluiden aikana ei myöskään löydetty syitä tai tarvetta käyttää muita ohjelmia graafiseen ja käyttöliittymäsuunnitteluun.

Adobe Photoshop on Adobe Systemsin kehittämä kuvankäsittelyohjelma, joka on suosituin kaupallisista kuvankäsittelyohjelmistoista. Monien muiden Adoben sovellusten tavoin Photoshop on saatavilla Mac OS:lle sekä Microsoft Windowsille. Emulaattoreita käyttäen ohjelmistoa voi myös ajaa muilla käyttöjärjestelmillä. Uusin versio ohjelmasta on nimeltään Photoshop Creative Suite 4 eli Photoshop CS4. (Adobe 2010).

Vaikka Photoshop onkin ensisijaisesti suunniteltu muokkaamaan kuvia painotuotantoon, sitä käytetään nykyään myös kuvien tekemiseen ja muokkaamiseen www-käyttöä varten. Käytimme Photoshoppia järjestelmämme ulkoasun suunnittelussa.

Photoshopin avulla suunniteltiin koko järjestelmän perusnäkymä ja sen komponenttien ulkoasu. Järjestelmä sisältää lukuisen määrän erilaisia elementtejä ja ikoneita, sekä kokonaisvaltaisen ja yhtenäisen ilmeen koko järjestelmälle, jota kaikkien järjestelmän eri osien tuli noudattaa. Muokkaamalla järjestelmälle täysin omanlaisensa ulkoasun voidaan keskittyä paremmin myös toiminnallisuuteen, koska järjestelmällä ei ole mitään valmiita määritelmiä eri komponenttien osalta. Järjestelmän käyttöliittymä yritettiin alusta alkaen rakentaa niin, että se olisi mahdollisimman helposti omaksuttava myös hieman kokemattomammallekin käyttäjälle.

Järjestelmän käyttöliittymän lisäksi tuli järjestelmään suunnitella kuvakkeet erityyppisille materiaaleille niin, että sisällön tyypin hahmottaminen olisi mahdollisimman helppoa. Erityyppisten sisältöjen kuvakkeet suunniteltiin käyttämällä Adobe Illustrator CS4 –vektorigrafiikkaohjelmistoa, joka on hyvin kattava vektorigrafiikkaan keskittyvä ohjelmisto. Pääasiassa Illustrator on tarkoitettu suuriresoluutioisten painomateriaalien ja muiden yksityiskohtaisten grafiikoiden suunnitteluun. (Adobe, 2009.) Koska kuvakkeet ovat kooltaan hyvin pieniä, mahdollisti Illustratorin käyttö hyvin yksityiskohtaisen graafikan tuottamisen.

Kuva 3. Esimerkkejä järjestelmään suunnitelluista havainnollistavista kuvakkeista

Käyttöliittymästä pyrittiin tekemään mahdollisimman käyttäjäystävällinen ja helppokäyttöinen. Järjestelmän eri osat pyrittiin sijoittelemaan selkeästi niin, että käyttäjän on helppo omaksua uusi käyttöliittymä nopeasti.

Kuva 4. Rautalankamalli järjestelmän ensimmäisestä käyttöliittymäsuunnitelmasta

Kehitettäessä käyttöliittymää tutkittiin aiemmin esiteltyjä valmiita materiaalinhallintaan käytettäviä ohjelmia ja käytettiin hyväksi niiden hyviä puolia. Yksikään tutkittu valmis ohjelma ei kuitenkaan vastannut täysin vaatimuksia joita kehitettävälle järjestelmälle asetettiin, joten muiden ohjelmistojen ominaisuuksia tuli mukauttaa paremmin oman projektin tarpeiden mukaisiksi, eikä suurta osaa ominaisuuksista voinut tai kannattanut käyttää sellaisenaan.

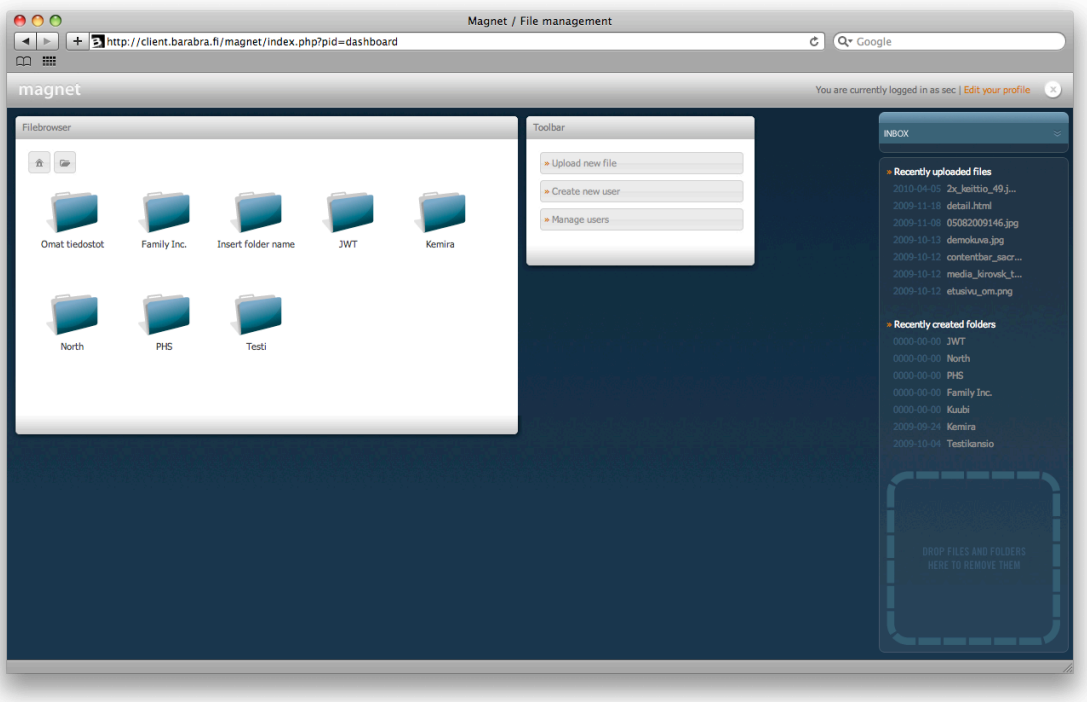
Käyttöliittymästä päätettiin toteuttaa työpöytämallinen, jossa käyttäjä voi itse vaikuttaa eri elementtien sijaintiin. Käyttöliittymä koostuu neljästä pääelementistä, jotka kaikki sijoitetaan niiden tarpeellisuuden ja tärkeyden mukaan loogisesti eri kohtiin käyttöliittymässä.

Yläpalkkiin sijoitettiin tietojärjestelmän logo sekä muutamia tarpeellisia perustoimintoja, joita käyttäjä ei tarvitse kokoajan. Tällaisia toimintoja ovat erimerkiksi siirtyminen omalle profiilisivulle ja uloskirjautumistoiminto, joka sijoitetaan loogiseesti oikeaan yläkulmaan.

Pääikkuna toimii näkymänä omiin kansioihin sekä tiedostoihin. Tässä ikkunassa voi navigoida järjestelmän sisällä kansioiden välillä. Tämä ikkuna sisältää myös tiedostojen tietoikkunan. Pääikkunaan rakennettiin oma navigaatiopalkki, johon sijoitettiin tärkeimmät ominaisuudet. Pääikkunan navigaation avulla käyttäjä pääsee mistä tahansa kansiosta takaisin pääkan-

sionäkymään, joten kansioita ei tarvitse selata turhaan päästäkseen takaisin alkuun. Navigaatioon sijoitettiin myös painike, josta käyttäjä pääsee edelliseen kansioon. Nämä toiminnot ovat kaikille tietokoneenkäyttäjille tuttuja heidän tietokoneensa tiedostojenhallinnasta, joten ominaisuuksien tarve myös järjestelmässämme oli selkeä.

Pääikkunan navigaation avulla käyttäjä voi myös luoda uusia kansioita sekä nimetä niitä. Pääikkuna tarkoitetaan materiaalien selaamiseen, muokkaamiseen, lisäämiseen sekä poistamiseen. Pääikkunaa käytetään yhdessä työkalupalkin kanssa, joka sijaitsee pääikkunan välittömässä läheisyydessä.



Kuva 5. Järjestelmälle suunniteltu lopullinen käyttöliittymä

Järjestelmään kehitettiin erillinen työkalupalkki, joka sijoitettiin erilleen pääikkunasta. Työkalupalkki sisältää tärkeimmät toiminnot jotka määräytyvät käyttäjän oikeuksien mukaan. Tästä syystä esimerkiksi kaikki oikeudet omaavalla käyttäjällä työkalupalkki näyttää erilaiselta kuin muilla käyttäjillä. Perusasetuksilla työkalupalkki asettuu päänäkymässä tiedostopalkin oikealle puolelle. Työkalupalkkia voi halutessaan raahata näkymässä vapaasti haluamalleen paikalle.

Kaikki oikeudet omaavalla käyttäjällä työkalupalkissa on mahdollisuus lisätä uusia tiedostoja, lisätä käyttäjiä, sekä selata ja poistaa käyttäjiä. Tällä käyttäjällä on oikeus lisätä tiedostoja mihin tahansa kansioon työkalupalkin avulla. Työkalupalkki on näkyvillä koko ajan, joten

kaikki tärkeimmät toiminnot ovat käytössä selasi käyttäjä pääikkunassa mitä kansiota tahansa.

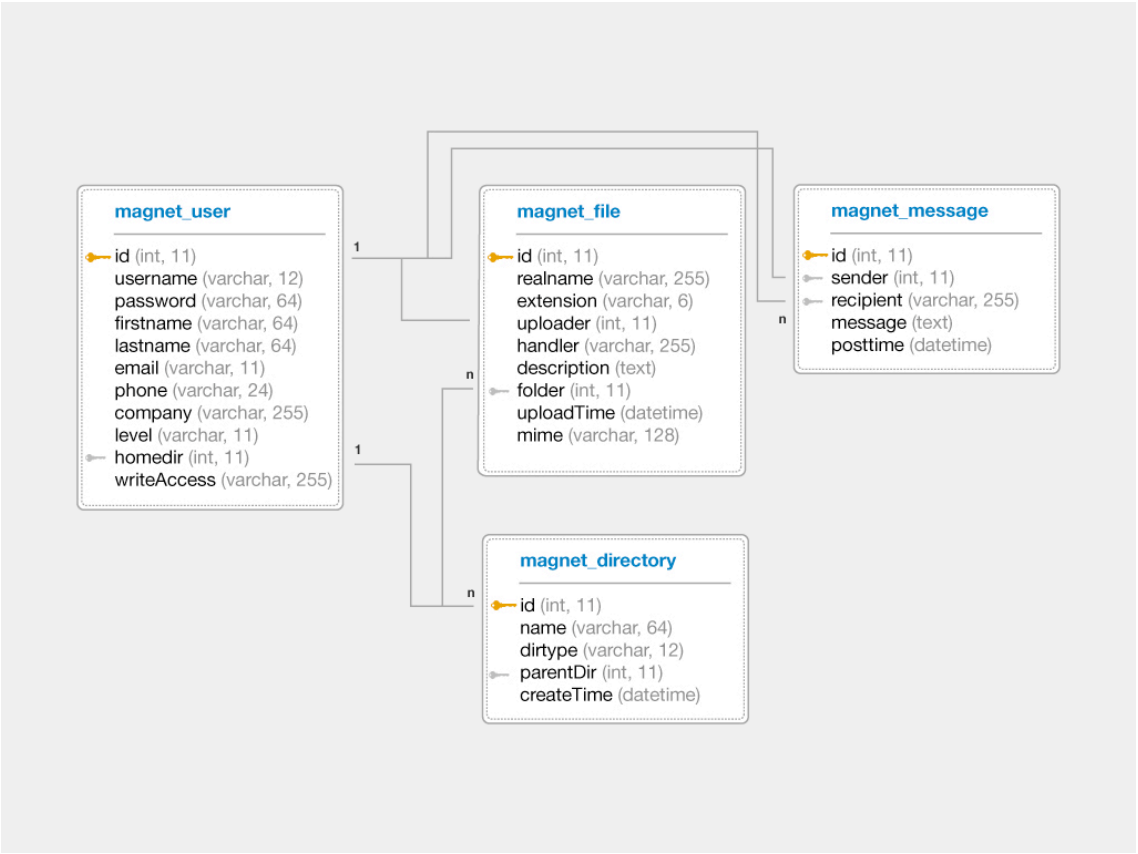
Tiedostojen poistamisesta haluttiin tehdä käyttäjälle mahdollisimman yksinkertaista, helppoa ja tehokasta. Päinvastoin kun vertailluissa järjestelmissä, päädyttiin siihen että tiedoston poistamiseen tarkoitettua toimintoa ei liitetty jokaiselle tiedostolle erikseen, vaan rakennettiin yksi yhteinen roskakori.

Kansioiden ja tiedostojen poistoon ei tehty mitään erillistä painiketta, vaan tiedostot ja kansiot voi poistaa ”drag and drop” –tyylisellä tekniikalla. Tekniikka on sama kuin yleisimmissä käyttöjärjestelmissä, joissa roskakoriin raahataan tiedostoja hiiren avulla. Roskakori sijaitsee aina järjestelmän käyttöliittymän oikeassa alakulmassa, joten se on aina käytettävissä.

Vahinkojen välttämiseksi roskakoriin tehtiin myös toiminto, että jos käyttäjä raahaa roskakoriin vahingossa väärän tiedoston tai kansion, järjestelmä kysyy käyttäjältä vahvistuksen. Näin käyttäjä voi vielä halutessaan peruuttaa tiedoston tai kansion poistamisen. Mikäli käyttäjä poistaa kansion, kaikki kansion sisällä sijaitsevat kansiot ja tiedostot poistuvat automaattisesti, joten käyttäjän ei tarvitse poistaa eri tiedostoja ja kansioita yksitellen, vaan kokonaisen kansion tuhoaminen alikansioineen on nopeaa.

10.2.5 Tietokanta

Järjestelmän tiedostojen tietojen ja tiedostorakenteen tallentamiseen käytetään MySQL-tietokantaa. Tietokanta rakentuu neljästä eri taulusta, jotka sisältävät tarvittavat solut tietojen tallentamiseen. Taulut on nimetty kuvaavasti kertomalla taulun sisällöstä sekä yhdistämällä taulut oikein on saatu aikaiseksi toimiva kokonaisuus materiaalinhallintajärjestelmän tarpeisiin. Oheisessa kaaviossa (Kaavio 9) on esitelty tietokannan rakenne ja taulujen väliset suhteet



Kaavio 9. MySQL –tietokannan rakenne sekä taulujen väliset suhteet

Kuten aiemmin todettiin, dokumentointi oli yksi tärkeä kokonaisuus projektissa. Dokumentaatioon panostettiin siksi, että järjestelmän mahdollinen jatkokehitys olisi erittäin helppoa. Tietokannan suunnitteluun ja mallintamiseen päätettiin käyttää näin ollen myös erillistä ohjelmaa, jolloin esimerkiksi tietokannan rakenteesta jäi talteen erittäin selkeät dokumentit. Näin jatkokehitys ja järjestelmään perehtyminen tulee olemaan paljon vaivattomampaa ja nopeampaa, vaikka järjestelmästä ei olisikaan aikaisempaa kokemusta. Seuraavaksi esitellään erikseen opinnäytetyön yhteydessä hyödynnettyjä tietokantasuunnitteluun tarkoitettuja ohjelmistoja.

MySQL Workbench on MySQL:n tekemä visuaalinen tietokantojen suunnittelutyökalu, joka on saatavilla useille eri käyttöjärjestelmille. Ohjelmisto tekee mahdolliseksi tietokantojen suunnittelun sekä näiden suunnitelmien siirtämisen suoraan tietokannaksi. (MySQL Workbench 2009.)

MySQL Workbench -ohjelmiston käyttöön päädyttiin sen kehittyneiden ominaisuuksien sekä suoran MySQL -yhteensopivuuden takia. Tietokannan taulut, kokonaisrakenne ja relaatiot suunnitteltiin tällä ohjelmistolla. Ohjelmisto mahdollisti tarkan taulu- ja relaatiosuunnittelun sekä auttoi hahmottamaan taulujen rakennetta yhtenä suurena kokonaisuutena. Tämän

lisäksi, toisin kuin muissa vaihtoehtoisissa ohjelmissa, MySQL Workbench:ssä oli mahdollista siirtää suunniteltu tietokantarakenne suoraan MySQL –tietokannaksi, eikä tauluja tarvinnut enää rakentaa käsin uusiksi. Nämä ominaisuudet tekivät MySQL Workbench:stä juuri tämän projektin suunnitteluun sopivan ohjelman.

Toinen tietokantakehityksessä hyväksikäytetty ohjelma oli Navicat. Se on graafisella käyttöliittymällä varustettu ilmainen ohjelma eri tietokantajärjestelmien selaamiseen ja muokkaamiseen. Navicatin avulla tietokantoja voi selata ja muokata ilman SQL-kyselyitä graafisen käyttöliittymän ja erilaisten työkalujen avulla. Navicatia käytettiin järjestelmän tuotantovaiheessa tietokannan hallintaan ja sillä seurattiin, että kehitysvaiheessa tieto verkkosovelluksesta tietokantaan siirrettäessä pysyi eheänä.

Projektiin tarkoitetulla internetpalvelimella oli valmiiksi asennettuna selainpohjainen phpMyAdmin-tietokantaohjelmisto, jota käytettiin myös samaan tarkoitukseen. Navicatin selkeämpi käyttöliittymä sekä tietokantataulujen helpompi ja nopeampi selattavuus tekivät Navicatista todella kätevän apuvälineen tietokantojen käsittelyyn, joten phpMyAdminia hyödynnettiin vain muutamassa tapauksessa. Navicat ei ole selainpohjainen toisin kuin phpMyAdmin, joten tietokantaa pystyi myös tarkastelemaan samaan aikaan kun itse järjestelmää testattiin selaimella. Koska Navicat on täysin ilmainen ohjelma, sitä voitiin käyttää järjestelmän kehitystyössä ilman lisäkustannuksia.

10.2.6 Tietoturva

Materiaalinhallintajärjestelmä sijaitsee fyysisesti Barabran tiedostopalvelimella. Palvelinta ylläpitää luotettavia verkkopaveluita tarjoava Nebula. Yksi avaintekijä ennen järjestelmän kehittämistä oli juuri tietoturva ja nykyisten käytäntöjen tietoturva-aukot, joista haluttiin eroon.

Uusi järjestelmä luo materiaalinhallintaan ja -jakamiseen täysin erilaiset puitteet kuin nykyinen järjestelmä, millä tarkoitetaan sähköpostia sekä avointa FTP-palvelinta. Uusi materiaalinhallintajärjestelmä sijaitsee palvelimella niin, että sen sisältämiin tiedostoihin on pääsy vain järjestelmän käyttöliitymän kautta, eikä tiedostoja palvelimelta voi ladata esimerkiksi syöttämällä tiedoston osoite selaimen URL –kenttään. Vähimmillääänkin tiedoston ladatakseen käyttäjän pitää tietää jokaiselle tiedostolle annettu “handler” (Kaavio 7), sekä sen MD5–tiivistesumma. Tämä lisää järjestelmän tietoturvallisuutta huomattavasti. Uusi järjestelmä estää siis kaikki haut niiltä käyttäjiltä, joilla ei ole tunnuksia järjestelmään.

Tietojärjestelmässä käytettiin tiedostojen ja salasanojen osalta MD5 -salausta. Salauksen myötä tietoturva paranee entisestään, koska se ehkäisee mahdollisia väärinkäyttötapauksia varsin tehokkaasti ja estää tietojen mahdollisen leviämisen.

Järjestelmään tallennettujen tiedostojen nimistä tallennetaan tietokantaan MD5-merkkijonot, joista ei voi tietää, minkä niminen tiedosto on kyseessä tai missä formaatissatiedosto on. Järjestelmä käsittelee tiedostoja viittaamalla tähän tarkistesummaan. Salausta käytetään myös tiedostojen nimissä tallennettaessa niitä tiedostopalvelimelle. Näin mahdollinen väärinkäyttäjäkään ei voi tietää palvelimelle päästessään mitä tiedostoja kansioissa on, tai minkälaisia tiedostot ovat.

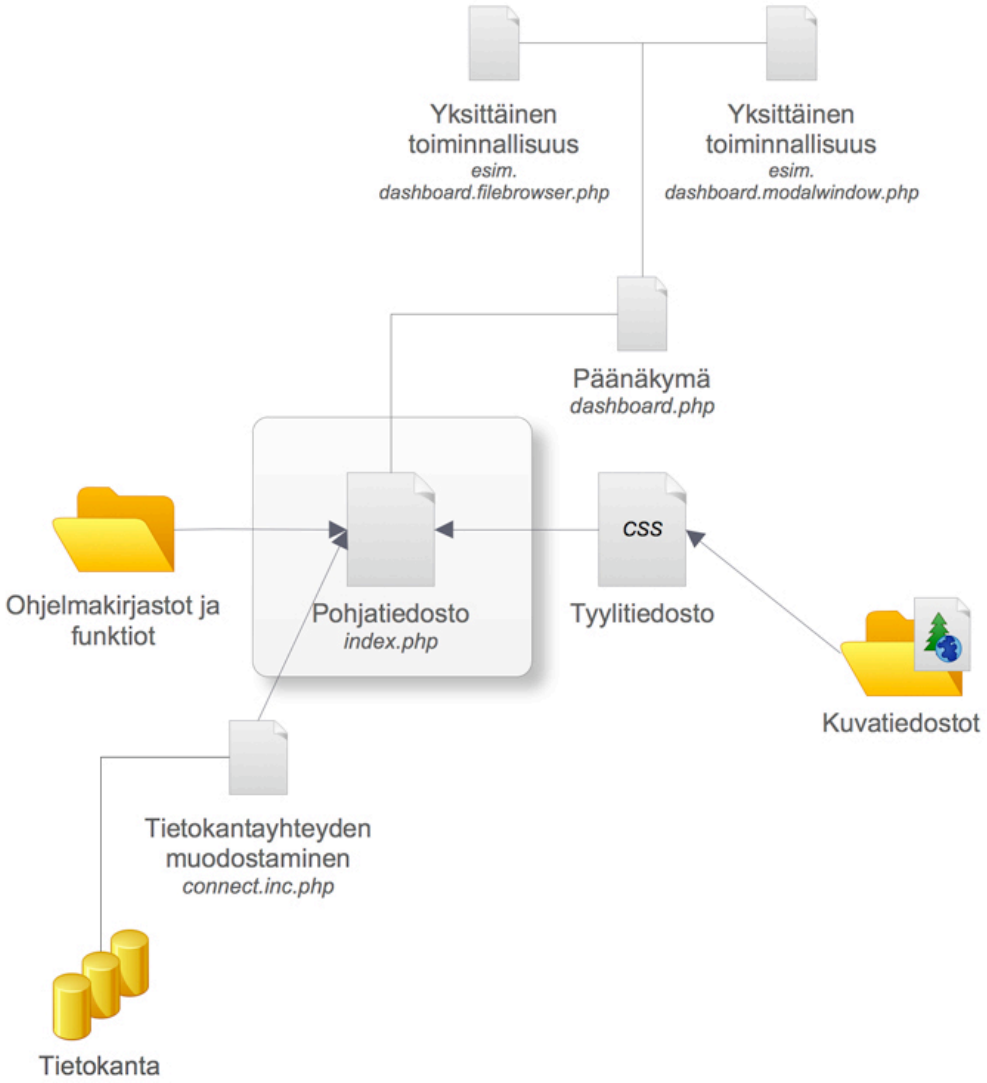
MySQL –tietokannan väärinkäyttö on suojattu sillä, että sitä ei voi käyttää kuin ainoastaan localhost –tunnuksilla tai suoraan materiaalinhallinan www-palvelimelimelta käsin. Eli tietokantaan ei pääse käsiksi edes asiakasohjelmistolla käyttämättä SSH -tunnelia.

10.3 Toteutus

Määrittelyn ja suunnittelun jälkeen aloitettiin järjestelmän toteuttaminen käyttöliittymän näkökulmasta. Käyttöliittymä tarvitsi erityishuomiota, koska oli selvitetty että järjestelmää tulisi käyttämään hyvin monipuolinen käyttäjäryhmä. Tästä syystä järjestelmän tuli olla hyvin käyttäjäystävällinen ja helposti sisäistettävä. Käyttöliittymän toteuttamisen jälkeen aloitettiin itse ohjelmistokehitys ja taustalle rakennettiin eri toiminnallisuudet sekä tietokannat. Tämän jälkeen suunniteltu käyttöliittymä integroitiin tähän taustatoiminnallisuuteen.

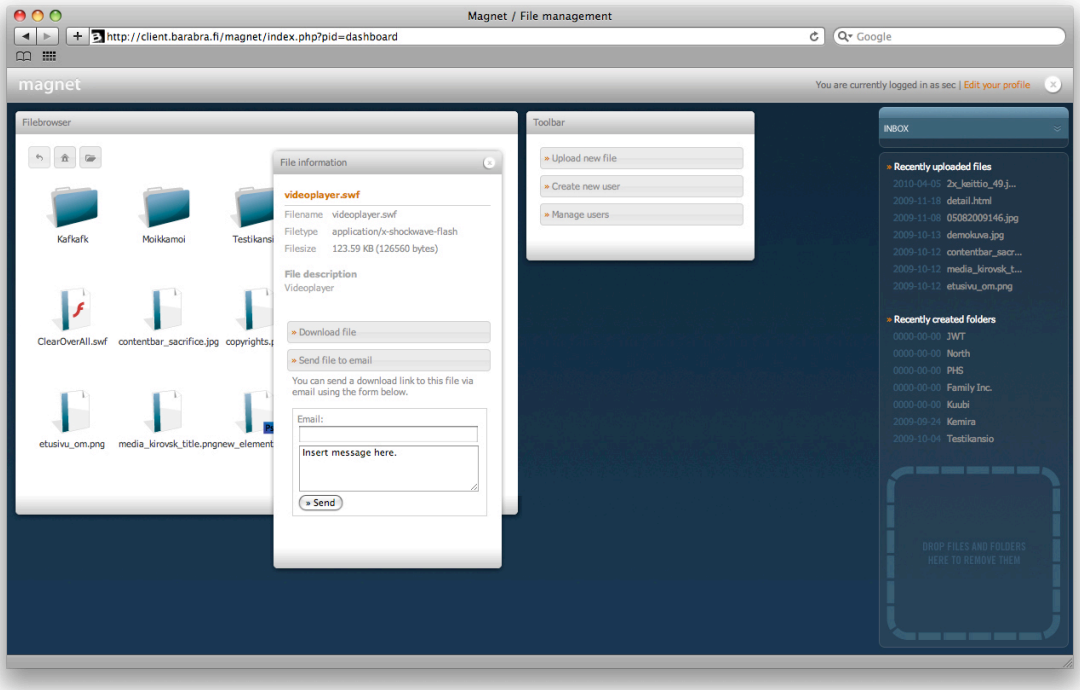
10.3.1 Käyttöliittymän toteuttaminen

Käyttölittymästä tehtiin www-sivupohjia (Liite 5) käyttäen HTML-merkkauskieltä sekä CSS-tyylitiedostoja (Liite 4). Teknisestä näkökulmasta käyttöliittymä toteutettiin niin, että päänäkymä rakentuu yhdestä sivupohjasta sekä tyylitiedostosta. Kaikki päänäkymään avautuvat ikkunat ja näkymät toteutettiin kukin omana sivupohjanaan, joista kaikki käyttivät yhtä yhteistä CSS-tyylitiedostoa. Toiminnallisuus eroteltiin erillisiin koodikirjastoihin (Liite 3) erilleen itse käyttöliittymästä (Kuva 4).

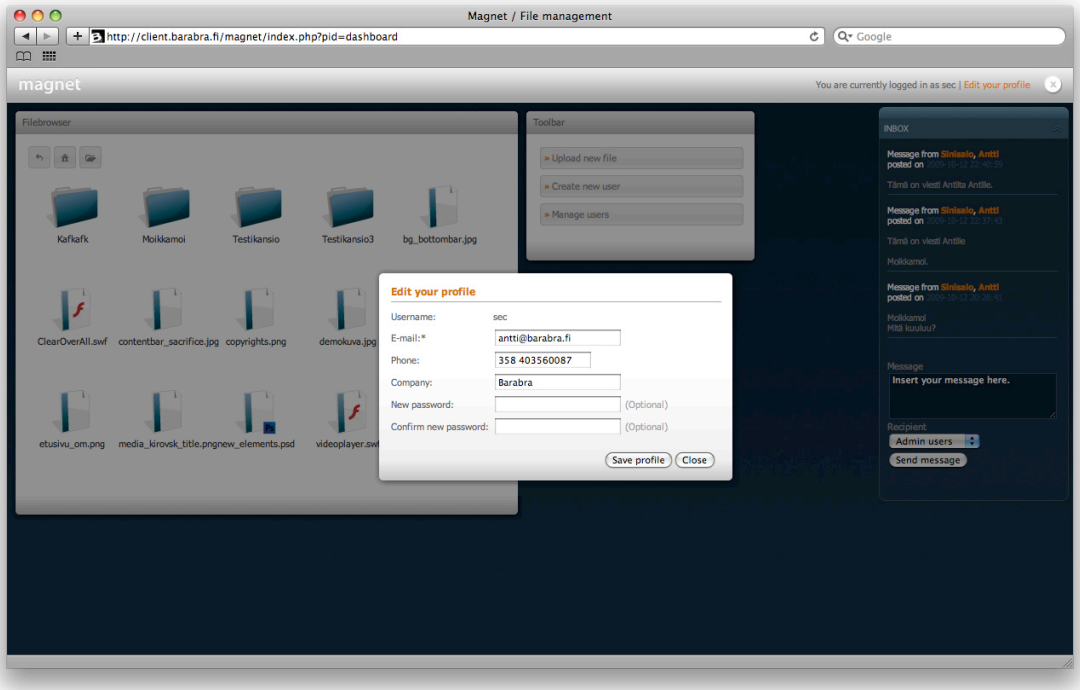


Kuva 6. Järjestelmän käyttöliittymän rakentumisperiaate

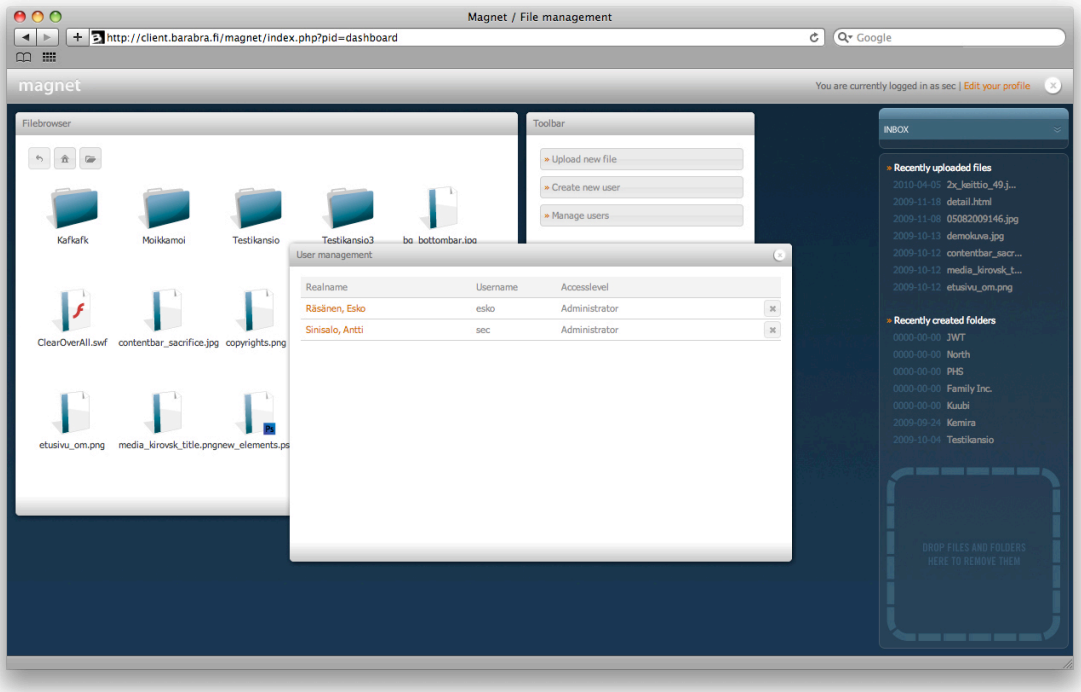
Aiemmin esiteltiin järjestelmän yleisilme ja päänäkymä (Kuva 5). Alla on esitelty järjestelmän eri osia kuvakaappauksin suoraan tilanteessa, jossa ulkoasu on työstetty jo osaksi järjestelmää.



Kuva 7. Näkymä järjestelmän tiedostojen esikatseluikkunasta



Kuva 8. Näkymä järjestelmän profiilinmuokkausikkunasta.

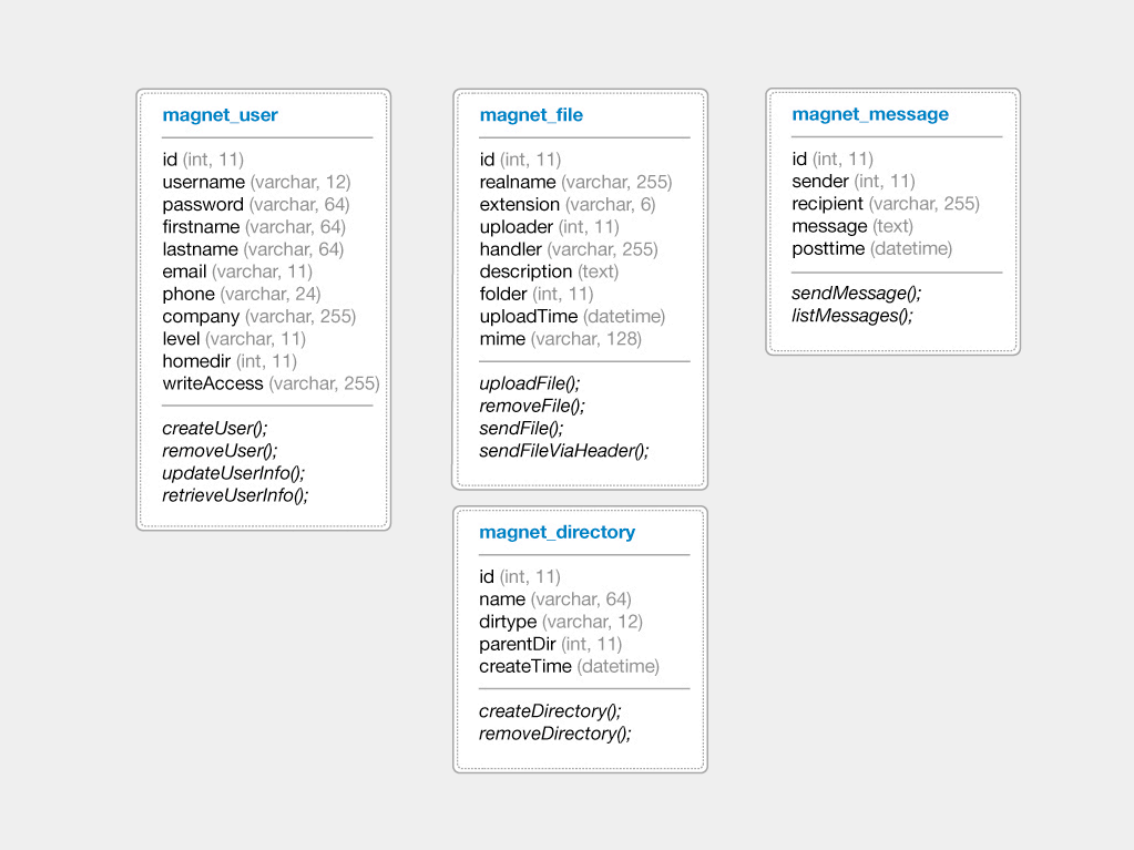


Kuva 9. Näkymä järjestelmän käyttäjienhallinnasta

10.3.2 Tietokannan ja sen taulujen rakentaminen

Järjestelmän taustalle rakennettiin tarvittavat tietokantataulut, joiden avulla ylläpidettiin tiedostokohtaisia tietoja, tiedostorakennetta sekä käyttäjien tietoja. Tiedokannat toteutettiin käyttäen Navicat -tietokantatyökalua.

Järjestelmä koostuu neljästä eri taulusta, joille kullekkin suunniteltiin niiden tietoja käsittelevät metodit. Alla kaavio, josta nämä luokat ja metodit käyvät ilmi. Opinnäytteeseen on myös liitetty dokumentti (Liite 6) josta käy ilmi jokaisen taulun SQL -kysely.



Kaavio 10: Järjestelmän luokkakaavio, josta käy ilmi eri luokat ja niiden metodit

”Magnet_User” –tauluun tallennetaan järjestelmän käyttäjien tiedot. Taulussa on solut käyttäjien perustiedoille, sekä lisäksi käyttäjän salasana, käyttöoikeudet sekä kotikansion tiedot. Tarkempi kuvaus taulusta on nähtävillä oheisesta kaaviosta (Kaavio 10), jossa on esitetty myös funktiot, jotka hakevat tietoa ja käsittelevät kyseistä taulua. Näitä funktioita ovat: createUser, removeUser, updateUserInfo sekä retrieveUserInfo. Edellä mainittuja funktioita käytetään käyttäjien lisäämiseen, poistamiseen, muokkaamiseen sekä hakemiseen.

”Magnet_file” -tauluun tallennetaan tiedot järjestelmään tallennetuista tiedostoista. Tiedostoista tallennetaan tiedot alkuperäisnimestä, tiedostopäätteestä, tiedostotyypistä, kansiosta, jossa tiedosto sijaitsee sekä ajankohdasta jolloin tiedoston on järjestelmään ladattu. Tarkempi kuvaus taulusta ja sen soluista on esitetty oheisessa kaaviossa (Kaavio 7), kuvassa on esitetty myös funktiot, jotka ovat yhdeydessä ”magnet_file” –tauluun.

”Magnet_directory” -tauluun tallennetaan tiedot järjestelmään luoduista kansioista. Kansioista tallennetaan tauluun tiedot kansion nimestä, tyypistä, emokansiosta sekä ajasta, jolloin kansio on luotu. Oheisessa kaaviossa (Kaavio 7) on tarkempi kuvaus taulun soluista sekä funktioista, joita käytetään luomaan uusia kansioita, sekä poistamaan kansioita.

”Magnet_message” –taulu tallentaa tiedot lähetetyistä viesteistä, joita käytetään tiedoston-latauslinkkien lähettämiseen. Tauluun tallennetaan viestin lähettäjä, vastaanottaja, viesti, sekä lähetysaika. Oheisessa kaaviossa (Kaavio 9) on tarkempi kuvaus taulun soluista, sekä funktioista, joita käytetään hakemaan tarvittavat tiedot taulusta.

10.4 Testaus ja sen suunnittelu

Ensimmäisen käyttövalmis järjestelmä otettiin koekäyttöön toimeksiantajayrityksessä heti sen valmistuttua. Järjestelmä päätettiin todeta toimivaksi siinä vaiheessa, kun se pystyy korvaamaan aiemmat materiaalinhallintaratkaisut lähes täydellisesti. Tähän tilanteeseen järjestelmä tulisi kuitenkin pääsemään vasta kun se on laajemmalti testattu myös asiakasyritysten puolesta. Muuttuvalla ja kiirellisellä toimialalla ei voida kuitenkaan olettaa, että järjestelmä pystyy tukemaan kaikkia mahdollisia käyttötapauksia.

Järjestelmätestauksen tarkoituksena on testata kohteen ominaisuus tai toimivuus ja varmistaa, että se vastaa järjestelmälle asetettuja vaatimuksia. Järjestelmätestauksen vaikeus vaihtelee suuresti testattavan sovelluksen monimutkaisuuden mukaan. Oikeellisuustestaus ja luotettavuuden testaaminen ovat testauksen kaksi suurinta osa-aluetta. (Jiantao Pan 1999.)

Taulukko 4. Tyypillisiä ohjelmiston laatutekijöitä (Jiantao Pan 1999)

Toiminnallisuus	Tekniikka	Muokattavuus
Oikeellisuus	Tehokkuus	Joustavuus
Luotettavuus	Testattavuus	Uudelleenkäytettävyys
Käytettävyys	Dokumentointi	Ylläpidettävyys
Yhtenäisyys	Rakenne	

Jo ennen kuin järjestelmästä oli valmiina lopullinen testiversio, sitä testattiin aika ajoin käymällä läpi kaikki toteutetut käyttötapaukset. Näin varmistuttiin siitä, että järjestelmän uudemmat ominaisuudet eivät olleet rikkoneet jo toteutettuja toimintoja.

Järjestelmän ensimmäisen testiversion valmistuttua se otettiin yrityksen sisäiseen testikäyttöön, jonka pääasiallisena tarkoituksena oli varmistaa sen toimivisuus eri käyttöalustoilla ja internet-selaimilla sekä tarkastaa ettei se sisältänyt suuria ongelmia sen käyttölogiikassa. Tämä testausryhmä keskittyi pääasiassa toiminnallisuuteen liittyviin tekijöihin, kuten järjestelmän luotettavuuteen. Koska testaus suoritettiin White-Box -testauksen mukaisesti, oli myös järjestelmän tehokkuus ja rakenne yksi testauksen keskeisistä kohteista. Päinvastoin kun Black-Box, jossa keskitytään vain testikohteen toimivuuteen, White-Box -testauksessa tutkitaan myös sitä, mitä tapahtuu järjestelmän sisällä. (TestingGeek, 2010.) Järjestelmän käytön

aikana yrityksen työntekijät raportoivat mahdollisista ongelmista ja virheistä joita he kokivat käyttäessään järjestelmää.

Testijakson edetessä kirjataan ylös mahdollisia puutteita, ongelmia tai tarpeellisia lisäominaisuuksia. Tämä vaihe auttaa ymmärtämään helpommin kuinka hyvin järjestelmä ratkaisee yrityksessä olleita materiaalinhallinnallisia haasteita, ja mitä järjestelmään täytyy lisätä, jotta ongelmat tulisivat varmasti paikatuksi. Mikäli ohjelmaa ei päätetä jatkokehittää käyttöönoton ja testauksen jälkeen, voidaan olettaa että ohjelma on teknisesti valmis ja toteuttaa kaikki sille annetut tavoitteet. Käytännössä kuitenkin järjestelmä on valmis vasta kun se siltä odotettu hyöty saavutetaan. Varsinkin juuri verkossa käytettävän ohjelman käyttönoton jälkeinen aika on käytettävä hyödyksi, koska kaikkia käyttötapauksia ei vielä aiemmin tunnettu. (Salmela 1999, 63.)

Kun järjestelmä oli testattu kohdeyrityksen sisäisessä käytössä ja ensimmäisessä testausvaiheessa todetut ongelmat oli korjattu, lisättiin testiryhmään muutama asiakasyritys. Asiakasyritykset pyrittiin valitsemaan niin että uusi testiryhmä kattoi eri käyttöympäristöjä, esimerkiksi eri käyttöjärjestelmäalustoja. Heille kerrottiin selkeästi mihin järjestelmän ominaispiirteisiin heidän tuli kiinnittää huomiota (käytettävyys, yhtenäisyys). Asiakasyritykset suorittivat testauksensa siis Black-Box -menetelmällä. Black-Box -testausmenetelmässä sovelluksen testaus perustuu sille annetujen syötteiden ja sen tuottamien tulosteiden oikeellisuuteen (Jiantao Pan 1999.)

Kun järjestelmää on testattu valitun käyttäjäryhmän avustuksella ja heidän raportoimansa ongelmat korjattu, otetaan järjestelmä käyttöön kaikissa projekteissa jossa se koetaan käytännölliseksi. Tässä vaiheessa voidaan sanoa että järjestelmästä on saatu aikaan ensimmäinen täysin valmis versio. Tämän jälkeen järjestelmää aletaan jatkokehittää lisäämällä sinne haluttuja uusia ominaisuuksia ja näin ollen voidaan testata myös sen muokattavuustekijöitä, kuten miten se mahdollisesti toimii päivitetyillä ohjelmistoversioilla.

11 Materiaalinhallintajärjestelmän jatkokehitys

Järjestelmän ensimmäisen käyttöönoton jälkeen oli selvää, että sen eri käyttäjäryhmät antavat palautetta. Tämän ulkopuolisten käyttäjien antaman palautteen lisäksi järjestelmää on tarkoitus kehittää jatkossa myös projektin aikana ilmi tulleiden ideoiden perusteella. Järjestelmän ominaisuudet rajattiin aluksi tarkkaan, jotta ensimmäinen julkaisu saadaan mahdollisimman aikaisin testikäyttöön. Järjestelmän jatkokehitys oli kuitenkin kokoajan tärkeässä osassa koko projektin etenemisen aikana, koska järjestelmästä haluttiin mahdollisimman joustava. Käyttäjälähtöisyyden takia on sitä kehitettävä aktiivisesti vastaamaan sen käyttäjien mieltymyksiä.

Alla on esitelty muutama aivoriihissä ja käyttäjäpalautteessa esiintullut kehitysidea, joita ei koettu välttämättömäksi ensimmäiseen julkaisuun. Nämä ideat ovat ovat kuitenkin ensimmäisen kehitysvaiheen muutoskierroksessa, jolloin ratkaistaan siihen asti esiin tulleita ongelmia ja parannusehdotuksia ja valitaan niistä tärkeimmät ja hyödyllisimmät.

11.1 Versionhallinta

Versionhallinta rajattiin pois ensimmäisestä vaiheesta, koska emme kokeneet sitä välttämättömänä. Versionhallinta helpottaa kuitenkin järjestelmän käyttöä ja tiedostojen selaamista. Yleensä tiedostojen nimeen liitetään jokin mahdollinen versionumero tai jokin muu määritelmä, mistä voi päätellä monesko versio kyseisestä tiedostosta kukin on.

Tähän ongelmaan pyritään kuitenkin löytämään jokin järjevä ratkaisu, ettei versionhallintaa täydy miettiä tiedostoa nimetessä ja mikäli järjestelmässä on jo vastaavan niminen tiedosto olemassa. Osittain ongelmaa ratkoo jo sivupalkissa näkyvä listaus uusimmista tiedostoista, josta voi ladata uusimmat lisätyt tiedostot. Sekään ei ratkaise ogelmaa kuin uusimman tiedoston kohdalla ja kun järjestelmässä on monta käyttäjää samaan aikaan, ei voida luottaa, että haluttu tiedosto näkyisi aina sivupalkissa.

Kunkin tiedoston esikatseluikkunassa olisi yksi mahdollisuus näyttää, mikäli tiedostosta on ladattavissa muitakin versioita. Tiedoston yhteydessä voisi olla esimerkiksi jonkin havainnollistava ikoni, joka ilmoittaa että tiedostosta on useita versioita ladattavissa. Mahdollisuuksia on monia, mutta tähän ongelmaan palataan varmasti jatkokehitysvaiheessa.

11.2 Käyttöliittymämuutokset

Yksi ongelma, joka mahdollisesti voi tulla vastaan lähitulevaisuudessa, on tiedostojen määrän lisääntyminen. Mikäli tiedostoja lisätään loogisesti eri asiakkaiden ja eri projektien alle, ei kansioiden sisällä olevien tiedostojen määrän tulisi kasvaa kovin suureksi. Joskus sellaisia projekteja kuitenkin on, että siirrettäviä tiedostoja on paljon, joten suurella todennäköisyydellä tämä saattaa koitua jatkossa ongelmaksi.

Tiedostojen paljous saattaa luoda käyttöliittymän ja nimenomaan tiedostojen selaamisen sekavammaksi. Testaamissamme järjestelmissä oli käytetty tiedostojen esittämisessä listaustyyliä, jossa tiedostot ovat alekkain, esitettynä pienillä ikoneilla. Tällöin selattavia tiedostoja mahtuu ruudulle enemmän ja selaaminen voi olla mielekkäämpää. Mikäli tiedostot esitetään eräänlaisena ruudukkona, kuten järjestelmän ensimmäisessä julkaisuversiossa, ikonit ovat isompia ja selkeämpiä, joten mikäli tiedostoja on vähän, on niitä helpompi selata ruudukkonäkymän avulla.

Tätä ongelmaa tarkastellaan mahdollisesti enemmän, kun järjestelmä on ollut jonkin aikaa käytössä ja voimme konkreettisesti nähdä, kuinka paljon tiedostojan ladataan järjestelmän eri projektien kansioihin ja onko käyttöliittymämuutokset tarpeellisia. Ongelma on kuitenkin olemassa ja siihen reagoidaan mikäli tarvetta on.

11.3 Esikatselu

Järjestelmään oli tarkoitus rakentaa jo ensimmäiseen vaiheeseen niin sanottu mediasoitin, jonka avulla käyttäjä voi katsoa järjestelmän sisällä olevan videotiedoston tai kuvan joutumatta lataamaan sitä koneelleen. Ominaisuus päätettiin kuitenkin jättää seuraavaan versioon, koska sitä ei koettu käytön kannalta välttämättömäksi.

Esikatselussa näkyy kaikki tarpeellinen, mitä tiedostosta tarvitsee tietää. Mediasoittimen avulla käyttäjä voisi kuitenkin tarkastella kuvia ja videoita huomattavasti helpommin, koska niitä ei tarvitse välttämättä ladata omalle koneelle. Varsinkin videotiedostot ovat usein hyvin raskaita, joten mitä vähemmän käyttäjän on ladattava niitä omalle koneelleen, sen parempi. Tämä helpottaa käyttöä varsinkin, mikäli kuvia ja videoita on paljon, ja jos niistä pitää tarkastella vain jotakin tiettyä. Tällöin ei ole juuri väliä, miten tiedostot on nimetty, vaan itse tiedoston sisältö kertoo itsestään enemmän, kuin sen tekniset tiedot. Varsinkin verkkomainosten, eli bannereiden osalta tämä on todella hyödyllinen ominaisuus, koska asiakas voi kommentoida mainosta lataamatta sitä omalle koneelleen.

12 Pohdinta

Opinnäytetyön tarkoituksena oli kehittää materiaalinhallintaa kohdeyrityksen ja asiakkaiden välillä. Tuotettavan järjestelmän tuli kohdentaa materiaalinvälitys yhteen paikkaan sen sijaan, että materiaaleja jaettaisiin sähköpostin ja tiedostopalvelimien kautta. Tavoitteena oli myös kehittää opinnäytetyön tekijöiden tietotaitoa tutkia olemassa olevia järjestelmiä sekä suunnitella ja toteuttaa tutkituista järjestelmistä saadun tiedon perusteella täysin uusi erillinen järjestelmä alusta loppuun.

Tietojärjestelmän kehitys sujui ilman suurempia ongelmia alusta loppuun. Molemmat opinnäytetyön tekijät ovat tuottaneet ennen opinnäytetyön tekemistä töissä hieman vastaavia järjestelmiä, joten hyvä pohja järjestelmän kehittämiselle oli jo olemassa. Käytetyt tekniikat olivat molemmille tekijöille ennestään enemmän tai vähemmän tuttuja, ja kun tekijät työskentelevät samassa yrityksessä ja tiedostivat omat ja toisen vahvuudet, oli myös työnjako selkeä alusta alkaen. Aluksi oli hieman epävarmaa, riittääkö kohdeyrityksellä tarpeeksi aikaa osallistua järjestelmän kehitystyöhön, mutta yhteistyö sujui kuitenkin vaivatta ja tarvittavat kes-

kustelut ja pohdinnat saatiin käytyä hyvin aikataulussa. Sisäiset projektit jäävät usein muiden jalkoihin, mutta nyt aikaa löytyi ja se vaikutti todella myönteisesti järjestelmän lopputulokseen sekä kehitysprosessiin.

Prosessin yksi parhaista puolista oli sen hyvin joustava aikataulu. Ylensä projektit viedään hyvin nopeasti alusta loppuun ja suunnittelulle ei varata kovin paljon aikaa. Nyt aikaa oli vähintäänkin riittävästi ja se näkyy lopputuloksessa. Tietojärjestelmä on testien palautteen perusteella jo tässä vaiheessa viimeistellyn oloinen, niin sisäisesti kuin ulkoisestikin. Tietojärjestelmä näyttää hyvältä, mutta on myös ohjelmakoodin puolesta jäsennelty hyvin selkeästi, sekä kommentoitu huolella, jotta varsinkin jatkokehitys olisi mahdollisimman helppoa. Vaikka järjestelmässä käytettävät tekniikat olivat kehittäjille jo ennestään tuttuja, osaaminen kehittyi projektin edetessä paljon osittain myös sille käytetyn ajan ansiosta. Kaikkia osa-alueita oli hyvin aikaa suunnitella ja toteuttaa. Erityisesti osaaminen kehittyi AJAX-tekniikan osalta, johon koko järjestelmä perustuu. Kyseinen tekniikka on nykyään hyvin suosittu, joten sen osaaminen on alalla on varsin tärkeää. Tästä syystä opinnäytetyön tekijöiden kannalta oli todella hyödyllistä toteuttaa koko järjestelmä AJAX-tekniikalla.

Aikaa projektin toteuttamiselle oli paljon, ja se auttoi osaltaan hyvin paljon projektin läpiviennissä, mutta toisaalta prosessin olisi voinut myös toteuttaa nopeammin. Yrityksen sisäisiin projekteihin ei kiinnitetä niin paljon huomiota ja tästä syystä sisäisien projektien aikataulut saattavat usein venyä hyvinkin pitkiksi. Aikataulullisesti projektin läpivientiprosessin olisi voinut suunnitella huomattavasti paremmin, jolloin jokainen osa-alue olisi saatu mahdollisesti toteutettua tehokkaammin. Liukuva aikataulu ei toisaalta millään tavalla haitannut järjestelmän toteutusprosessia. Tämä on ehkä suurin kehityskohde tämän projektin osalta, ja se luo myös paljon kysymyksiä yrityksen sisäisiä projekteja kohtaan. Mikäli sisäisiin projekteihin kiinnitettäisiin enemmän huomiota ja toteutettaisiin tehokkaammin, jäisi myös muolle projekteille enemmän aikaa.

Kohdeyritykselle ja asiakkaille järjestelmä on testikäyttöönoton jälkeen osoittautunut varsin onnistuneeksi. Konkreettisesti järjestelmän julkaisun näkee sähköpostien määrässä, koska suurin osa materiaalista jaetaan nyt järjestelmän kautta. Ennen järjestelmän käyttöönottoa materiaalit saattoivat tulla hyvinkin monessa eri sähköpostiviestissä, jolloin materiaalin hallinta ja etsiminen jälkeenpäin hankaloitui. Nyt kaikki materiaalit on haettavissa ja jaettavissa ajasta riippumatta yhdessä paikassa, eikä niitä tarvitse pyytää erikseen sähköpostilla. Kaiken kaikkiaan projekti sujui hyvin, varsinkin järjestelmästä saadun palautteen perusteella, jota tulee yllättävän paljon. Järjestelmän ensimmäinen versio tuntuu jo tehneen tehtävänsä, mutta tulevaisuudessa järjestelmästä pyritään kehittää vielä edistyneempi ja käyttökelpoisempi.

Lähteet

About MySQL. 2009. Luettu 22.7.2009.
http://www.mysql.com/about/

Adobe: Adobe Illustrator CS4. 2009. Luettu 22.7.2009
http://www.adobe.com/products/illustrator/

Adobe: Photo Editing Software. Luettu 27.04.2010.
http://www.adobe.com/products/photoshop/family/

Arvidsson, S. & Ek, J. 1999. Tietokantojen käsittely Internetissä. Vantaa: Schildts Kustannus Oy – Pagina (Tummavuoren Kirjanpaino Oy).

Asleson, R. & Scutta. N.T. 2006. Ajax – Tehokas hallinta. Jyväskylä: Gummerus.

Crichlow, M. 2001. Hajautetut tietojärjestelmät. Helsinki: Edita Oyj.

Ek, J. & Norén, K-J. 1999. Dynaaminen HTML käytännössä. Vantaa: Schildts Kustannus Oy – Pagina (Tummavuoren Kirjanpaino Oy).

Helsingin Yliopisto: Aivoriihi. Luettu 19.5.2010.
http://www.valt.helsinki.fi/blogs/opetuskesk/aivoriihi.htm

IBM Rational Unified Process. Luettu 27.04.2010.
ftp://ftp.software.ibm.com/software/rational/web/datasheets/RUP_DS.pdf

Jiantao Pan, Software Testing, 1999. Luettu: 24.4.2010.
http://www.ece.cmu.edu/~koopman/des_s99/sw_testing/

jQuery Project. 2009. Luettu 27.7.2009.
http://jquery.org/history

Kruchten, P. 2003. The rational unified process: an introduction. Boston: Pearson education, Inc.

MySQL Workbench: About MySQL Workbench. 2009. Luettu 16.7.2009.
http://wb.mysql.com/?page_id=6

Parkkinen, J. 2002. Hyvään verkkopalveluun! Helsinki: Inforviestintä Oy.

Pelto-Piri, J. 2009. Tuottajan kanssa käyty keskustelu. Barabra Oy.

Salmea, J. 1999. Nettijärjestelmän rakentaminen. Jyväskylä: IT Press.

Talentum. 2005. Tietojärjestelmän hankinta: ohjelmistotoimittajan ja –ratkaisun valinta. Jyväskylä: Gummerus Kirjapaino.

TestingGeek, 2010. Luettu: 24.4.2010.
http://www.testinggeek.com/index.php/testing-types/system-knowledge/50-white-box-testing

tSoft: Vaatimusmäärittely. Luettu 19.05.2010.
http://cs.joensuu.fi/tSoft/vaatimusmaarittely.htm

Vilkka, H. 2007. Tutki ja mittaa – Määrällisen tutkimuksen perusteet. Jyväskylä: Gummerus kirjapaino.

Valtionvarainministeriö: Käyttäjälähtöisyys verkkopalveluiden suunnittelussa. 2008. Luettu: 27.4.2010.
http://www.vm.fi/vm/fi/04_julkaisut_ja_asiakirjat/01_julkaisut/04_hallinnon_kehittaminen/20080129Kaeyttae/verkkopalveluiden_suunnittelu.pdf

What is MD5?. 2009. Luettu 16.6.2009.
http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci527453,00.html

W3Schools: Browser Statistics. 2009. Luettu 20.8.2009.
http://www.w3schools.com/browsers/browsers_stats.asp

Liitteet

Liite 1 Sisäänkirjautumista käsittelevä ohjelmakoodi

```
<?php
include_once("../common/config_inc.php");
// checkLogin.php
// functions for validating the user and creating corresponding cookies

                // Retrieve incoming variables
                $magnet_username=md5(mysql_real_escape_string($_POST['magnet_username']));
                $magnet_password=md5(mysql_real_escape_string($_POST['magnet_password']));

                // dbQuery for username and password validation
                $result = mysql_query("SELECT * FROM magnet_user WHERE
md5(username)='$magnet_username' and password='$magnet_password' ORDER BY id") or
die(mysql_error());

                $num_rows = mysql_num_rows($result);


                if($num_rows==1)
                {
                // dbQuery results in a valid user, set cookies
                $expire=time()+60*60*24*30;
                setcookie("magnet_username", $magnet_username, $expire, "/", COOKIEDOMAIN);
                setcookie("magnet_password", $magnet_password, $expire, "/", COOKIEDOMAIN);
                setcookie("magnet_loggedIn", "true", $expire, "/", COOKIEDOMAIN);

                header("location: ../".ROOT."?pid=dashboard");
                } else  {
                // dbQuery results in an invalid user, do not set cookies, display error.
                header("location: ../".ROOT."?err=login:1");
                }

?>
```

Liite 2 Järjestelmän päänäkymän pohjustus

```
<?php
include_once("common/config_inc.php");
include_once("common/common.validateCookies.php");
include_once("common/common.byteSize.php");
include_once("common/common.errorHandler.php");
include_once("common/common.retrieveUserInformation.php");

$contentPointer = $_GET['pid'];
if($contentPointer=="" || !isset($contentPointer)) $contentPointer="login";
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Magnet / File management</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<link type="text/css" href="assets/styles/mainstyle.css" rel="stylesheet" >
<link type="text/css" href="assets/styles/jquery-ui-1.7.2.custom.css" rel="Stylesheet" />

<script type="text/javascript" src="assets/js/jquery.latest/jquery-1.3.2.min.js"></script>
<script type="text/javascript" src="assets/js/jquery.latest/jquery-ui-1.7.2.custom.min.js"></script>
<script type="text/javascript" src="assets/js/jquery.pngfix/jquery.pngfix.js"></script>

<script type="text/javascript">
$(document).ready(function(){
                $(document).pngFix();
                });

                $(function() {
                $("#accordion").accordion({ header: "h3" });
                });

                function startUpload() {
                document.getElementById('upload_process').style.visibility = 'visible';
                return true;
                }

                function stopUpload(success, dirId){
                var result = '';
                if (success == 1){
                                document.getElementById('uploadResult').innerHTML =
                                '<center><span class="msg">File was uploaded successful-
ly!<\/span></center>';

                                openDirectory(dirId);
                                retrieveLatestUploads();
                                } else {
                                document.getElementById('uploadResult').innerHTML =
                                '<center><span class="emsg">File with same filename already
                                exists.<\/span></center>';
                                }

                                document.getElementById('upload_process').style.visibility = 'hid-
den';
                                return true;
                }

                function slideIn(target){
                var options = {};
                options = { to: {width: 200,height: 60} };
                $(target).toggle("blind",options,500);
                };
                </script>

</head>
<body>

<div id="container">
    <div id="main">
                <?php if($_COOKIE['magnet_loggedIn']=="true") include("common/layout.header.php"); ?>
                <?php include("content/".$contentPointer.".php"); ?>
    </div>
</div>

</body>
</html>
```

Liite 3 Käyttöliittymätoimintoihin liittyvää ohjelmakoodia

```
// Functions to handle user-interface actions

$("#toolbar_button_send").disableSelection();
$("#toolbar_button_upload").disableSelection();

$("#createFail").hide();
$("#createSucceed").hide();
$("#createFillError").hide();

$("#updateFail").hide();
$("#updateSucceed").hide();
$("#updateFillError").hide();

$("#messageFail").hide();
$("#messageSucceed").hide();

                $("#updateUserButton").click(function(){


                $("#updateFail").hide();
                $("#updateSucceed").hide();
                $("#updateFillError").hide();

                var email = $("#upduserEmail").val();
                var phone = $("#upduserPhone").val();
                var company = $("#upduserCompany").val();
                var password1 = $("#upduserPassword").val();
                var password2 = $("#upduserPassword2").val();

                if(password1=="" && password2=="" && email!="" && phone!="")  {
                                updateUser(email, phone, company, password1, password2);
                                return false;
                                }
                                else if(password1!="" && password2!="" && password1!=password2) {
                                $("#updateFail").fadeIn();
                                $("#updateSucceed").hide();
                                $("#updateFillError").hide();
                                }
                                else if(email=="" || phone=="")
                                {
                                $("#updateFail").hide();
                                $("#updateSucceed").hide();
                                $("#updateFillError").fadeIn();
                                }
                                else {
                                updateUser(email, phone, company, password1, password2);
                                return false;
                                }
                });

                function updateUser(email, phone, company, password1, password2){
                var _vars = "email="+email+"&phone="+phone+"&company="+company+"
                &password1="+password1+"&password2="+password2;
                $.ajax({
                                type: "POST",
                                url: "functions/updateUser.php",
                                data: _vars,
                                dataType: "text",
                                success: function(success) {
                                                if(success) {
                                                                $("#updateFail").hide();
                                                                $("#updateFillError").hide();
                                                                $("#updateSucceed").fadeIn();
                                                } else {
                                                                $("#createSucceed").hide();
                                                                $("#createFillError").hide();
                                                                $("#createFail").hide();
                                                                }
                                                }
                                });
                }

                $("#createUserButton").click(function(){
                                var username = $("#newuserName").val();
                                var email = $("#newuserEmail").val();
                                var phone = $("#newuserPhone").val();
                                var firstname = $("#newuserFirstname").val();
                                var lastname = $("#newuserLastname").val();
                                var company = $("#newuserCompany").val();
                                var access = $("#newuserAccess").val();
                                var writeAccess = $("").val();

                                if ($('#newuserWrite:checked').attr('checked')) {
                                                var writeAccess = true;
                                } else {
                                                var writeAccess = false;
                                }

                                if(username!="" && email!=""){
                                                createNewUser(username, email, phone, firstname,
                                                lastname, company, access, writeAccess);
                                                return false;
                                } else {
```

```
                                $("#createFail").hide();
                                $("#createSucceed").hide();
                                $("#createFillError").fadeIn();
                }

});

function createNewUser(username, email, phone, firstname, lastname, company,
access, writeAccess){
                var _vars = "username="+username+"&email="+email+"&phone="+phone+"
                &firstname="+firstname+"&lastname="+lastname+"&company="
                +company+"&access="+access+"&writeAccess="+writeAccess;
                $.ajax({
                                type: "POST",
                                url: "functions/createNewUser.php",
                                data: _vars,
                                dataType: "text",
                                success: function(success) {

                                if(success){
                                                $("#createFail").hide();
                                                $("#createFillError").hide();
                                                $("#createSucceed").fadeIn();

                                                // Reset the form field values if the
                                                user wants to add another user
                                                $("#newuserName").val("");
                                                $("#newuserEmail").val("");
                                                $("#newuserPhone").val("");
                                                $("#newuserFirstname").val("");
                                                $("#newuserLastname").val("");
                                                $("#newuserCompany").val("");
                                                $("#newuserAccess").val("");
                                } else {
                                                $("#createSucceed").hide();
                                                $("#createFillError").hide();
                                                $("#createFail").fadeIn();

                                }
                                }
                });
}

$("#sendMessageButton").click(function(){
                var recipient = $("#messageRecipient").val();
                var message = $("#messageMessage").val();
                var sender = $("#messageSender").val();

                if(message!="" && message!="Insert your message here."){
                                sendMessage(message, recipient, sender);
                                return false;
                } else {
                                $("#messageFail").fadeIn();
                                $("#messageSucceed").hide();
                }

});

function sendMessage(message, recipient, sender){
                var _vars = "messa
                ge="+message+"&recipient="+recipient+"&sender="+sender;
                $.ajax({
                type: "POST",
                url: "functions/sendMessage.php",
                data: _vars,
                dataType: "text",
                success: function(success) {

                                if(success) {
                                                $("#messageFail").hide();
                                                $("#messageSucceed").fadeIn();

                                                // Reset the form field values if the
                                                user wants to send another message
                                                $("#messageMessage").val("Insert your
                                                essage here");
                                } else {
                                                $("#messageSucceed").hide();
                                                $("#messageFail").fadeIn();
                                                }
                                }
                });
}

// Make dropbox DOM element droppable
$(".dropbox").droppable({
                accept: '#files > div',
                activeClass: 'dropbox_active',
                hoverClass: 'dropbox_om',
                drop: function(event, ui) {
                                var answer = confirm("Are you sure you want to remove
                                the selected file?")
                                if (answer) {
                                removeItem(ui.draggable);
                                }
                }
});
```

```
// Function to handle item removal
function removeItem($item) {
                $.ajax({
                                url : "func
                                tions/removeItem.php?itemId="+$item.attr('id'),
                                success : function (data) {
                                                openDirectory(data);
                                                }
                                });
}

// Make DOM elements draggable
$(function() {

                $("#filebrowser").resizable({
                                maxHeight: 750,
                                maxWidth: 600,
                                minHeight: 250,
                                minWidth: 600,
                });

$("#filebrowser").draggable({ containment: 'parent', handle: 'p', opacity: 0.7 });
$("#filebrowser_drag").disableSelection();
$("#toolbar").draggable({ containment: 'parent', handle: 'p', opacity: 0.7 });
$("#toolbarr_drag").disableSelection();

$.ajax({
                url : "functions/listDir.php?dir="+<?php echo retrieveUserInformati
                on("rootdir");?>,
                success : function (data) {
                                $("#filebrowser_content").html(data);
                                }
                });

});

// Retrieve list of latest messages
function retrieveMessages(){
$.ajax({
                url : "functions/getMessages.php",
                success : function (data) {
                $("#messenger_content").html(data);

                }
});
}

retrieveMessages();

// Slide in and out elements in toolbar and messenger window
$(function() {

$("#toolbar_content1").hide();
$("#toolbar_content2").hide();
$("#messenger_container").hide();

$("#messenger_button").click(function() {
                slideIn("#messenger_container");


                if($("#messenger_button").attr("src")=="assets/images/messenger_top_c
                losed.png") {
                                $("#messenger_button").attr("src",
                                "assets/images/messenger_top_open.png");
                                $("#rightbar").fadeOut();

                } else {
                                $("#messenger_button").attr("src",
                                "assets/images/messenger_top_closed.png");
                                $("#rightbar").fadeIn();
                }
return false;
});

                $("#toolbar_button_upload").click(function() {
                                slideIn("#toolbar_content1");
                                return false;
                });

                $("#toolbar_button_createuser").click(function() {
                                slideIn("#toolbar_content2");
                                return false;
                });
});

// Retieve list of recently uploaded files and display it on right bar.
function retrieveLatestUploads()
{
$.ajax({
                url : "functions/latestUploads.php",
                success : function (data) {
                                $("#rightbar_latestfiles").html(data);
                }
});
}
```

```
                // Retrieve list of recently created folders and display it on right bar.
                function retrieveLatestFolders()
                {
                $.ajax({
                                url : "functions/latestFolders.php",
                                success : function (data) {
                                                $("#rightbar_latestfolders").html(data);

                                }
                });
                }

                retrieveLatestUploads();
                retrieveLatestFolders();

                $('#toolbar_button_manageusers').click(function() {
                                openUserManagement();
                })

                function openUserManagement()
                {

                $.ajax({
                                url : "functions/listUsers.php",
                                success : function (data) {
                                                //$('#fileInfo'+fileId).html(data);

                                                $('#wrapper').prepend(data);
                                                $('#userbrowser').draggable({ containment: 'parent',
                                                handle: 'p',stack: { group: "*",
                                                min: 50 }, opacity: 0.7});
                                                $('#userbrowser_drag').disableSelection();
                                }
                });

}


// Modal window activity
$(document).ready(function() {

    $('a[name=modal]').click(function(e) {
        e.preventDefault();
        var id = $(this).attr('href');

        var maskHeight = $(window).height()-$('#header').height();
        var maskWidth = $('#main').width();

        $('#mask').css({'width':maskWidth,'height':maskHeight});

                        $('#mask').fadeIn(800);

        var winH = $(window).height();
        var winW = $(window).width();

        $(id).css('top',  winH/2-$(id).height()/2);
        $(id).css('left', winW/2-$(id).width()/2);

        $(id).fadeIn(800);

    });

    $('.window .close').click(function (e) {
        e.preventDefault();
        $('#mask, .window').hide();
    });

    $('#mask').click(function () {
        $(this).hide();
        $('.window').hide();
    });

});
```

Liite 4 Katkelma käyttöliittymän tyylitiedostosta

```
@charset "UTF-8";
/* CSS Magnet*/

body, html {
                margin-top: 0px;
                margin-bottom: 0px;
                margin-left: 0px;
                margin-right: 0px;
                padding: 0px;
                height: 100%;
                background-color: #162c40;
                font-family: tahoma;
                font-size: 11px;
                color: #585858;
                overflow: hidden;
}

.red {
                color: #FF3300;
}

.small {
                font-size: 10px;
}

.title {
                font-size: 11px;
                color: #CC6600;
                font-weight: bold;
                border-bottom: 1px solid #CCC;
                height: 24px;
}

.tdrowtitle {
                height: 20px;
                color: #999;
}

#container {
                position:relative;
                height: 600px;
                min-width: 1100px;
}

#header {
                z-index: 999;
                height: 41px;
                background-image: url('../images/bargrad.jpg');
                clear:both;
}

#headerContent {
                position: absolute;
                top: 13px;
                right: 50px;
                float: right;
                border: 0px solid #fff;
}

#headerContent a {
                color: #CC6600;
                text-decoration: none;
}

#headerContent a:hover {
                color: #999;
                text-decoration: none;
}

#logo {
                clear: none;
                float: left;
}

#logout {
                clear: none;
                float: right;
}

#main {
                background-image: url('../images/bggrad.jpg');
                background-repeat: repeat-x;
                min-height: 100%;
                height: 100%;
                overflow: hidden;
}
```

```
#wrapper {
                position: relative;
                clear: none;
                width: 83%;
                height: 100%;
                float: left;
                overflow: hidden;
}

#loginBox {
                position: relative;
                width: 345px;
                height: 166px;
                background-image: url('../images/loginBox.png');
                margin-left: auto;
                margin-right: auto;
                top: 200px;
}

#loginBoxTextContent {
                position: absolute;
                top: 58px;
                left: 38px;
}

#loginBoxError {
                position: absolute;
                clear: none;
                top: 114px;
                left: 38px;
}

#loginBox a {
                color: #a65e01;
                font-size: 9px;
                text-decoration: none;
}

#loginBox a:hover {
                color: #c56f00;
                font-size: 9px;
                text-decoration: none;
}

.invisibleInputfield {
                border: 0px solid #fff;
                background-color: #ebebeb;
                font-family: tahoma;
                font-size: 11px;
                color: #666666;
}

#filebrowser {
                clear: none;
                float: left;
                width: 600px;
                height: 300px;
                padding: 0.5em;
                position: relative;
                padding-bottom: 80px;
}
```

Liite 5 Tiedostonselaus -ikkunan sivupohja

```
<div id="filebrowser" class="ui-widget-content">
<div id="filebrowser_drag">
                <div id="filebrowser_header_topleft">
                                <img src="assets/images/window_topleft.png" border="0" />
                </div>
                <div id="filebrowser_header">
                                <p class="filebrowser_header_text">Filebrowser</p>
                </div>
                <div id="filebrowser_header_topright">
                                <img src="assets/images/window_topright.png" border="0" />
                </div>
</div>
                <div id="filebrowser_content"></div>
                <div id="filebrowser_footer"></div>
</div>
```

Liite 6 Tietokantataulujen SQL -kyselyt

```
CREATE TABLE `magnet_user` (
  `id` int(11) NOT NULL auto_increment,
  `username` varchar(12) NOT NULL,
  `password` varchar(64) NOT NULL,
  `firstname` varchar(64) NOT NULL,
  `lastname` varchar(64) NOT NULL,
  `email` varchar(64) NOT NULL,
  `phone` varchar(24) NOT NULL,
  `company` varchar(255) NOT NULL,
  `level` varchar(11) NOT NULL,
  `homedir` int(11) NOT NULL,
  `writeAccess` varchar(255) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;



CREATE TABLE `magnet_file` (
  `id` int(11) NOT NULL auto_increment,
  `realname` varchar(255) NOT NULL,
  `extension` varchar(6) NOT NULL,
  `filesize` int(255) NOT NULL,
  `handler` varchar(255) NOT NULL,
  `description` text NOT NULL,
  `folder` int(11) NOT NULL,
  `uploadTime` datetime NOT NULL,
  `mime` varchar(128) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;



CREATE TABLE `magnet_directory` (
  `id` int(11) NOT NULL auto_increment,
  `name` varchar(64) NOT NULL,
  `dirtype` varchar(12) NOT NULL,
  `parentDir` int(11) NOT NULL default '0',
  `createTime` datetime NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;



CREATE TABLE `magnet_message` (
  `id` int(11) NOT NULL auto_increment,
  `sender` int(11) NOT NULL,
  `recipient` varchar(255) NOT NULL,
  `message` text NOT NULL,
  `posttime` datetime NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Liite 7 Projektiaikataulu