WWW-SISÄLLÖNHALLINTAJÄRJESTELMIEN SUUNNITTELU JA KÄYTTÖ

LAHDEN AMMATTIKORKEAKOULU
Tekniikan ala
Mediatekniikan koulutusohjelma
Teknisen visualisoinnin suuntautumisvaihtoehto
Opinnäytetyö
10.5.2010
Mikko Nuuttila

Lahden ammattikorkeakoulu
Tekniikan laitos
Mediatekniikan koulutusohjelma

Nuuttila, Mikko: Www-sisällönhallintajärjestelmien suunnittelu ja käyttö.

Teknisen visualisoinnin opinnäytetyö, 68 sivua, 0 liitesivua

Kevät 2010

TIIVISTELMÄ

Opinnäytetyössä käsitellään www-sisällönhallintajärjestelmien käyttöä ja suunnittelua. Työssä selvitetään, millaisia järjestelmiä on tarjolla tämän hetken markkinoilla ja millaisia eroavaisuuksia näiden järjestelmien välillä on. Markkinatutkimuksen tarkoituksena on antaa hyvä pohja oikean järjestelmän valinnalle.

Järjestelmien tutkinnalla ja toteutuksella pyritään ennen kaikkea löytämään ratkaisuja erityylisiä organisaatioita varten. Suuri painoarvo annetaan käyttäjäystävällisyyteen ja pitkänäköiseen sisällönhallintatoteutukseen. Nopeasti muuttuvaa sisällönhallintajärjestelmien markkinaa tutkitaan käyttämällä hyödyksi lähteitä, joiden informaation ajankohtaisuus on taattu. Tällaisia lähteitä ovat mm. web-statistiikkaa tarjoavat sivustot.

Opinnäytetyössä on kaksi CASE-projektia: CASE-projekteina on kaksi web-sivua, joiden sisällönhallintaa valitaan toisistaan poikkeavat lähestymistavat. Toinen järjestelmistä toteutetaan valmiilla avoimen lähdekoodin järjestelmällä ja toista varten suunnitellaan itse oma järjestelmä.

Avainsanat: sisällönhallinta, www-sisällönhallinta, www-sisällönhallintajärjestelmä, julkaisujärjestelmä, sivupohja, sisällönhallinta suunnittelu

Lahti University of Applied Sciences
Faculty of Technology

Nuuttila, Mikko: Web content management systems design and use

Bachelor´s Thesis in Vizualition Engineering, 68 pages

Spring 2010

ABSTRACT

This thesis deals with the design and use of web content management systems. The thesis describes what systems are currently available and what the differences between these systems are. The aim of this market research was to provide a good basis for selecting the right system for different purposes.

The objective was foremost to find solutions for different styles of organizations. In the analysis, high priority was given to user-friendliness and long-sighted content management implementation. The rapidly changing market for content management systems was studied using sources that have guaranteed up-to-date information, for example web-sites that provide statistics for other web-sites.

The thesis has two case projects. They are two web pages, which both have a content management system, but the approach is different in both pages. One site was built using a ready-made open source system, and the other site has it own customized system.

Keywords: content management, web content management, web content management, publishing, templates, content management design

SISÄLLYS

1. JOHDANTO

Www-sisällönhallintajärjestelmät auttavat web -sivujen ylläpitäjiä ja luovat web-sivuista toimivan kokonaisuuden eri laajuisia organisaatiota varten. Sivujen sisältömäärän kasvaessa järjestelmän käyttöönotosta tulee välttämätöntä. Aihealueena www-sisällönhallintajärjestelmät on vielä hyvin nuori, ja tästä johtuen siihen liittyvät markkina-alueet ovat vielä melko epäkypsiä.

Tässä opinnäytetyössä keskitytään tutkimaan erityyppisten www-sisällönhallintajärjestelmien toimintaa ja sitä kuinka näitä järjestelmiä käytetään. Järjestelmien tutkiminen aloitetaan tutkimalla asiantuntijoiden näkemyksiä järjestelmien toiminnoista ja käsittelemällä poikkeavia tulkintoja aihealueeseen liittyen. Käsitteiden ja termien kattava ymmärtäminen tulee olemaan välttämätöntä opinnäytetyön selkeyden ja johdonmukaisuuden kannalta. Perehtymällä syvällisesti järjestelmien toimintaa siirrytään tutkimaan tällä hetkellä tarjolla olevia järjestelmiä, josta saatua tietoa käytetään hyödyksi sopivinta järjestelmää valittaessa.

Vallitsevia markkinoita tutkittaessa järjestelmät jaetaan avoimen lähdekodin järjestelmiin ja kaupallisiin järjestelmiin. Tällä kahtiajaolla pyritään selkeyttämään näiden kahden järjestelmätyypin ominaisuuksien esittelyä. Erottelun avulla opinnäytetyössä pystytään myös vertailemaan näiden kahden tyypin vahvuuksia ja heikkouksia.

Opinnäytetyöhön sisältyy kaksi case-projektia joiden tarkoituksena on soveltaa opinnäyteyön teoriaosuutta ja havainnollistaa tässä esitettyjä tietoja. Projektien esittelyssä keskitytään valitun toteutustyylin yleisiin toimintatapoihin esittämällä teoreettista tietoa aiheesta. Tällä tavoin pyritään vähentämään teknisen toteutuksen huomiota ja keskitytään tutkimaan projektiin liittyvää aihetta laaja-

alaisemmin. Molempiin projekteihin sisältyy täysimittaisen web -sivun luonti, mutta opinnäytetyössä keskitytään ainoastaan näiden sivujen sisällönhallintatoteutuksiin. Toisen sivun sisällönhallinta toteutetaan avoimen lähdekoodin Joomla-järjestelmällä, ja toista sivua varten luodaan itse räätälöity ratkaisu sisällönhallintatoimintoja varten. Oman järjestelmän toteutuksen kuvauksessa keskitytään suunnitteluun ja toimivan kokonaisuuden rakentamiseen eikä varsinaiseen ohjelmointitoteutukseen.

2. WWW-SISÄLLÖNHALLINTA

2.1 Sisällönhallinta

Sisällönhallinta määritellään tieteenalana prosessien, teknologian, konseptien ja ratkaisujen joukkona, joilla on tekemistä sisällön kehittämisen, keräämisen, hallinnoinnin ja julkaisun kanssa (Friedllein, 2003). Sisällönhallinta on aiheena monijakoinen, joten aiheeseen liittyvät määritelmät saattavat erota toisistaan merkittävästi. Sisällönhallintajärjestelmä tarkoittaa eri ihmisille eri asiaa riippuen siitä mitä hän tekee ja mikä on hänen osuutensa sisällönhallintaprosessissa. Sisällönhallinnan merkityserot organisaation eri osille tulevat hyvin esiin Boikon (Boiko, 2005) luettelemissa esimerkeissä.

Liiketoiminnan näkökulmasta sisällönhallinta antaa taloudellista tukea, analyyttisistä näkökulmista sisällönhallinta tasoittaa organisoitavia voimavaroja, ammatillisesti katsottuna sisällönhallinta yhdistää sisältöpohjaisia tiedonaloja, tutkiessa työprosessia sisällönhallinta kerää, hallinnoi ja julkaisee informaatiota ja teknisestä näkökulmalta sisällönhallinta on toimiva tekninen infrastruktuuri. Listatuilla esimerkeillään Boiko pyrkii havainnollistamaan sisällönhallinnan merkitystä organisaatiossa ja sitä, kuinka laaja-alainen sisällönhallinta on aihealueena.

Organisoitava sisältö voi olla tekstimuotoisia dokumentteja, kuvatiedostoja tai www-sivuja. Yleisin ja nopeimmin kasvava sisällönhallintamuoto on tällä hetkellä www-sisällönhallinta. Www-sisällönhallinnalla tarkoitetaan yleisesti toimintaa, jolla organisoidaan verkkopalvelujen sisältöä mahdollisimman tehokkaasti. Www-sisällönhallinnalle on tyypillistä sisällön, rakenteiden ja ulkoasun erottaminen omiksi osikseen. Sivupohjat (eng. Template) määrittelevät sivun ulkoasun, joka voidaan määritellä sopivaksi eri medioita varten. Eri kohderyhmille muokattavissa olevan sisällön ansiosta www-sisällönhallinta liitetään usein monikanavajulkaisuun.

Goodwin ja Vidgen toteavat (2002), että www-sisällönhallinnan perustoiminta on suhteellisen helppo ymmärtää, mutta tarkemman määrittelyn antaminen on kuitenkin haastavaa, koska www-sisällönhallinnan merkitystä muihin aihealueisiin ja tietojärjestelmiin ei voida jättää huomioimatta. Goodwin ja Vidgen esittävät www-sisällönhallinnan olevan kokonaisuus kolmesta erillisestä osa-alueesta: dokumenttien ja työkulun hallinnasta, ohjelmistokehityksen versionhallinnasta sekä asiakkuudenhallinta-ohjelmistoista ja verkkokauppasovelluksista.

2.2 Www-sisällönhallinta prosessirakenteena

2.2.1 Tuotantolinjamalli

Www-sisällönhallintaa on hyvä ajatella prosessina, joka seuraa monivaiheista työkulkua. Tutkimalla tätä vaiheittaista prosessia saadaan selvillä kuinka sisällönhallintajärjestelmä toimii ja mitä asioita tulee huomioida kunkin vaiheen aikana. Voidaankin sanoa, että prosessirakenne tuo esiin sisällönhallinnan ominaisuudet ja vastuut. Työkulun tunteminen on erittäin arvokasta valitessa oikea sisällönhallintajärjestelmää organisaatiolle tai uutta järjestelmää luodessa. Sisällönhallinnan prosessin jakamisesta ja esittämisestä on useita eri tulkintoja. Tarkkuus, jolla prosessin vaiheet esitetään, johtaa useimmissa tapauksissa tulkintojen poikkeavuuksiin.

Tyypillisesti www-sisällönhallintaprosessi kuvataan suorana tuotantolinjamaisena mallina, jossa prosessin eri vaiheiden jälkeen siirrytään säännönmukaisesti seuraavaan. Kuva 1. on mukaelma Friedlenin sisällönhallintaprosessista. Kaaviossa esitetään myös aihealueita ja toimintoja, jotka ovat tyypillisiä työkulun eri vaiheille. Friedlein jakaa www-sisällönhallinnan kolmeen osaan: kerääminen, hallinnointi ja julkaisu. Työkulun aloittaa keräämisvaihe, jossa informaatio kerätään järjestelmää varten oikeassa muodossa ja rakenteessa. Sisällön kirjoittajalta saatava tieto on muutettava sopivaan muotoon järjestelmää varten, minkä jälkeen tätä sisältöä käyttämällä voidaan luoda järjestelmään sopiva sisältö-objekti, johon tarvit-

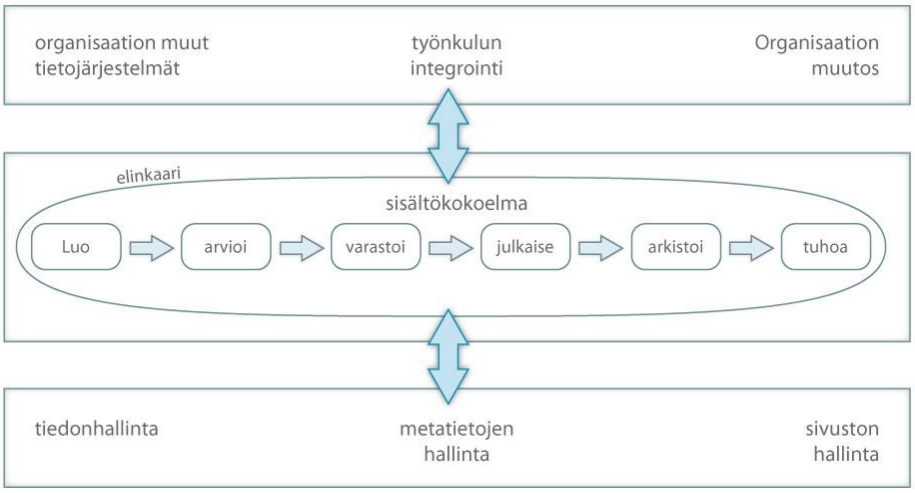
taessa lisätään myös metadataa. Prosessimallin seuraavassa vaiheessa hallinnoidaan järjestelmään lähetettyä tietoa. Sisällön sijoittamisen jälkeen järjestelmään tulee järjestelmän myös tarjota mahdollisuudet mm. etsiä, järjestää, versioida ja linkittää tietoa. Tuotantolinjamallin viimeinen osa, julkaisu sisältää kaiken mitä tarvitaan sisällön tuomiseen ulos järjestelmästä julkiseen käyttöön. Julkaisun oleellisimmat osat ovat käyttöönotto ja ulkoasun määrittelevä sivupohja (template). Käyttöönotolla tarkoitetaan uuden sisällön varmistettua julkaisua. Oikea sisältö halutaan julkaista oikeaan aikaan tai halutaanko sisältö julkaista välittömästi järjestelmään siirron jälkeen. Julkaistulle sisällölle tulee myös tarjota mahdollisuus personointiin määriteltyjen käyttäjien kohdalla. Sivupohjalla määritellään julkaisun ulkoasu, ja sillä erotellaan sisältö itse sivun ulkoasusta.

Kuva 1. Friedleinin tuotantolinjamalli prosessikuvauksesta

2.2.2 Goodwin ja Vidgenin moniulo tteinen malli

Goodwinin ja Vidgenin prosessimalli on jaettu Friedleinin mallin tavoin kolmeen osaan. Kuva 2. pohjautuu Goodwinin ja Vidgenin malliin työprosessikuvauksesta. Nuolet merkitsevät työkulun suunnan prosessimallissa. Malli ei ole kuitenkaan samalla tavoin suoralinjainen vaan toimintojen suunta riippuu työkulun tavoittees-

ta. Goodwin ja Vidgen ovat myös huomioineet prosessien eri osien sisäisen työkulun ja sen merkityksen. (Goodwin & Vidgen, 2002)



Kuva 2. Työprosessin kuvaus (mukaillen Goodwin ja Vidgen, 2002)

Kuvion ylin osa kuvaa organisaation organisaation tietojärjestelmiä tämänhetkisiä työnkulkua ja organisaatioilta odotettavaa muutosta. Keskimmäinen osa on kuvion keskeisin osa-alue. Tämä alue sisältää varsinaisen toiminnon sisällönluomiselle esittäen samalla sisällön elinkaaren. Tähän alueeseen sisältyy myös sisältökokoelma, joka on kanava kaikkiin organisaation eri tietokantoihin ja tiedostojärjestelmiin. Goodwind ja Vidgen painottavat elinkaaren merkitystä, joka jatkuu luomisesta tuhoamiseen asti. Tällaisella mallilla pyritään luomaan kaikkein käytännönläheisin esimerkki siitä miten sisältöä käsitellään järjestelmässä. Ennen julkaisua uusi sisältö saattaa vaatia arvioinnin. Tällä vaiheella pyritään varmistamaan onko uusi sisältö hyväksyttävää tallennettavaksi. Julkaisuvaiheeseen sisältyy sisällön muuttaminen järjestelmään sopivaan muotoon. Julkaisun jälkeen sisältö voidaan varastoida, jolloin sisältöön voidaan lisätä mm. metadataa tai muuta hyödyllistä tietoa, jota saatetaan tarvita jatkossa. Tämä vaihe voi tapahtua täysin automaattisesti tai manuaalisesti järjestelmästä riippuen. Viimeisenä vaiheena on sisällön tuhoaminen, joka tapahtuu määriteltynä ajankohtana. Sisältökokoelma voi sisältää tietokantoja sekä tiedostoja. Sisällönhallintajärjestelmällä tulee olla suora yhteys näihin kaikkiin osiin, jotta sisällön elinkaari olisi toimiva.

Alin osa koostuu tiedonhallinnasta, metadatan hallinnasta ja sivuston hallinnasta. Tiedonhallinnalla tarkoitetaan sisältöjen tallentamista järjestelmään sen omilla menetelmillä. Goodwin ja Vidgen suosittelevat käytettäväksi XML-tiedostoihin pohjautuvia toimintoja tiedon tallentamiseen ja lukemiseen mm. tiedostojen hierarkkisen rakenteen takia. XML:ää käytetäänkin monissa www-sisällönhallintajärjestelmissä vaikka suurin osa sisällöstä tallennetaan tietokantoihin. Metadata:n avulla sisältöön voidaan luoda lisämääritteitä, jotka auttavat sisällön käytössä. Tällaisia lisämääritteitä voivat olla mm. avainsanat, luontiajankohta ja versionumero. Sivuston hallinnalla tarkoitetaan järjestelmän osuutta, jossa sisältö erotetaan ulkoasusta.

Vertaillessa Goodwinin ja Vidgenin prosessikuvausta Friedlenin versioon voidaan molemmista kuvauksista löytää tiettyjä yhtäläisyyksiä. Goodwinin ja Vidgenin kuvauksen keskeisin osa-alue sisältökokoelman työnkulku vastaa hyvin pitkälle Friedlenin kokonaista prosessikuvausta. Goodwinin ja Vidgenin esitysmallin suurin ero onkin tavassa tutkia aihetta. Heidän mukaansa www-sisällönhallinnan ei tulisi olla irrallinen järjestelmä organisaation muista tietojärjestelmistä. He ovat myös huomioineet prosessikuvauksessa organisaation muun toiminnan. Avarakatseisempi lähestymistapa prosessin kuvaukseen tekee sen esittämisestä monimutkaisemman ja vaikeasti tulkittavan, mutta huomattavasti realistisemman käytännön toimintoja ajatellen.

2.3 Julkaisujärjestelmä vai sisällönhallintajärjestelmä

Asiantuntijat käyttävät molempia otsikossa mainittuja termejä käsitellessä www-sisällönhallintaa varten kehitettyjä järjestelmiä. Asian konteksti määrittää paljon, kumpaa nimie voidaan käyttää tai kumpi termeistä on tässä yhteydessä tarkempi. Molemmat termeistä on yleisesti käytössä, joten näen tärkeänä, että määrittelen, kuinka itse lähestyn tutkimaan tätä asiaa opinnäytetyössäni.

Julkaisujärjestelmillä tarkoitetaan usein järjestelmää joka käsittelee valtaosin sisällön lopullista tasoa julkaisua. Keskeisimmät ominaisuudet näissä järjestelmissä on WYSIWYG-editori ja tähän liittyvät muotoilutoimenpiteet. Mahdollisuutta laajempaan työkulun hallintaan tai versiohallintaan ei ole. Perttu Tolvanen määrit-

telee artikkelissaan sisällönhallintajärjestelmäksi järjestelmän, jonka ominaisuudet liittyvät juuri sisällönhallintaan. Yksinkertaisemmat kotisivujen päivitystyökalut, kuten kotisivukone tai putteri, ovat lähempänä julkaisujärjestelmää. (Perttu Tolvanen, 2008, vierityspalkki.fi). Omassa opinnäytetyössäni puhun kuitenkin myös tämän tason järjestelmistä sisällönhallintajärjestelmä nimikkeellä aihealueen selkeyttämisen vuoksi. Erotan kuitenkin ylläpitotyökalut kuten DreamWeaverin tästä ryhmästä. Eräiden tulkintojen mukaan järjestelmä voi koostua pelkästään näistä työkaluista ylläpitotoimintoja tekevän henkilön kanssa.

2.4 WWW-sisällönhallintajärjestelmät

Kaikilla organisaatioilla, joilla on web-sivut, tulee olla myös www-sisällönhallintajärjestelmä. Tämä järjestelmä voi koostua vain yhdestä sivuston ylläpitäjästä, joka toimii täysin manuaalisesti, tai käytössä voi olla kehittyneempi järjestelmä, jonka avulla tuotetaan ja julkaistaan sisältöä hienostuneemman tietojärjestelmän avulla. (Goodwin ja Vidgen, 2002, 66). ”Tietojärjestelmä” saattaa joissain tapauksissa olla ainoastaan joukko erityyppisiä kehitystyökaluja. Toisissa tapauksissa käytössä on yksittäinen järjestelmä, jonka avulla hoidetaan kaikki ylläpitotoimenpiteet. Www-sisällönhallintajärjestelmillä pääsääntöisesti tarkoitetaankin tämäntyylisiä järjestelmiä, ja näihin järjestelmiin keskitytään tässä opinnäytetyössä.

Www-sisällönhallintajärjestelmät (eng. Web content management tai web CMS) ovat tyypiltään julkaisupainotteisia, ja usein www-sisällönhallintajärjestelmistä puhutaankin www-julkaisujärjestelmä nimikkeellä. Tunnusomaista www-sisällönhallintajärjestelmille ovat pienet itsenäiset sisältöyksiköt ja tasomainen rakenne. Järjestelmän rakenne on rakennettu useaan eri tasoon, jotka ovat järjestetty hierarkkisesti. Ylin taso vaikuttaa suoraa kaikkiin muihin tasoihin ja ylläpitäjä, jolla on suurin mahdollinen käyttöoikeus sivustoon, hallinnoi tämän tason sisältöä ja toimintoja. Ylimmät tasot sisältävät tietoja, kuten sivun templaterakenteen, toiminnalliset funktiot ja sivuston meta-tiedon. Alimmilla tasoilla on dynaamisempaa aktiivisesti päivitettävää sisältöä, kuten tekstisisältö ja irralliset liitetiedostot.

2.5 Www-sisällönhallintajärjestelmien toimintaperiaate

2.5.1 Web-toteutuksen eroavaisuudet ja termien käyttö

Www-sisällönhallintajärjestelmien toimintaperiaatetta on hyvä lähteä tarkastelemaan tutkimalla eritasoisia, tyylisiä www-julkaisutapoja. Useimmat www-sisällönhallintajärjestelmät toimivat samojen perusperiaatteiden mukaan, mutta täysin yleispätevän esimerkin luominen järjestelmien toiminnasta on hyvin vaikeaa havainnollistaa mm. edellisessä kappaleessa mainittujen tietojärjestelmien poikkeavuuksien takia. Goodwinin ja Vidgenin tulkinnan mukaan myös staattiset web-sivut voivat sisältää sisällönhallintajärjestelmän, koska järjestelmän toimintamalli voi pohjautua ainoastaan yhteen manuaalisesti toimivaan ylläpitäjään. (Goodwin ja Vidgen, 2002). Seuraavissa kappaleissa tarkastellaan www-sisällönhallintajärjestelmien toimintaa Boikon mallien mukaan, jossa staattiset ja dynaamiset sivut erotaan sisällönhallintajärjestelmiä käyttävistä web-sivuista. Tämä tulkinta vastaa vallitsevia tulkintoja, ja se lähestymistapa on sopivin tähän opinnäytetyöhön.

2.5.2 Staattiset web-sivut

Staattinen web-sivu on joukko HTML-tiedostoja ja niihin liittyviä tiedostoja, jotka on varastoitu erillisesti web-serverille. Staattiset sivut ovat rakenteeltaan ja toiminnoiltaan yksinkertaisimpia. Niiden sisältö ja ulkoasu on ennalta rakennettua, eivätkä ne sisällä mahdollisuuksia personointiin. Muutokset ja päivitykset on tehtävä sivuille siis täysin manuaalisesti jokaiselle sivulle erikseen. Staattisia sivuja käytetään valtaosin mm. pienissä yrityksissä, joilla ei ole tarvetta aktiiviseen sivujen päivittämiseen. Hyötynä staattisilla sivuilla on niiden toimintanopeus, koska staattisia sivuja ladatessa ei tarvitse ajaa lainkaan ylimääräisiä prosesseja. (Boiko, 2005, 75.)

2.5.3 Dynaamiset web -sivut

Dynaamisia sivuja kutsutaan myös tietokantapohjaisiksi sivuiksi. Niissä web-sivu luodaan käyttäjän selaimelle suoraa esitetyn kyselyn mukaan. Sivujen sisältö on tallennettu tietokantoihin tai XML-tiedostoihin. Sivujen ulkoasu pohjautuu sivupohjiin (engl. Template), jotka sisältävät tavallista HTML:ää, CSS:ää sekä toiminnallisia ohjelmointi-skriptejä ja muita ohjelmia, jotka tulkitsevat sivujen käyttäjän toimintoja. Sivujen kyselyä vastaava sisältö haetaan ja yhdistetään sivupohjaan, josta luodaan tarvittava HTML-tiedosto selaimelle. Täysin dynaamisena web-sivuna voidaan pitää sivuja, joka ei sisällä laisinkaan HTML-tiedostoja vaan ainoastaan kyvyn luoda niitä. Dynaamisia sivuja pidetään usein sisällönhallintajärjestelmällisinä sivuina, mutta on kuitenkin huomioitava, että tällä toimintaperiaatteella toimiva www-sivu ei tee laisinkaan varsinaista sisällönhallintaa tai sisällä mitään siihen liittyviä tyypillisiä toimintoja, joita on käsitelty aiemmissa kappaleissa. (Boiko, 2005, 75-77.)

2.5.4 Sisällönhallintajärjestelmän sisältävä web-sivu

Sisällönhallintajärjestelmän sisältävä web-sivu toimii hyvin samalla periaatteella kuin dynaaminen web-sivu. Sivu koostuu osittain dynaamisesta ja osittain staattisesta sivu-rakenteesta. Boiko on listannut asioita, joita sisällönhallintajärjestelmällä toimiva web-sivu voi sisältää ja mitä toimintoja näihin kuuluu.

* Sisällönhallintasovellus: Sovelluksen tarkoituksena on kerätä ja hallinnoida sivujen sisältöä ja työkulkua sekä mahdollistaa ylläpitotoimenpiteet. Sovelluksen arkkitehtuuri vaihtelee tuotteittain. Joissain organisaatioissa järjestelmä saattaa toimia sisäisen verkon kautta ja toisissa tapauksissa yleisellä web-serverillä.

* Tietolähteet: Serverillä on oma paikka tallennetulle sisällölle, ylläpitäjien tiedoille ja muulle sisällölle, jota järjestelmä tarvitsee sivujen toteuttamiseen. Näitä tietolähteitä käytetään järjestelmän toimintaa varten eikä tuo-

maan varsinaista sisältöä itse sivuille. Tieto voi olla talletettu tietokantoihin tai XML-tiedostoihin.

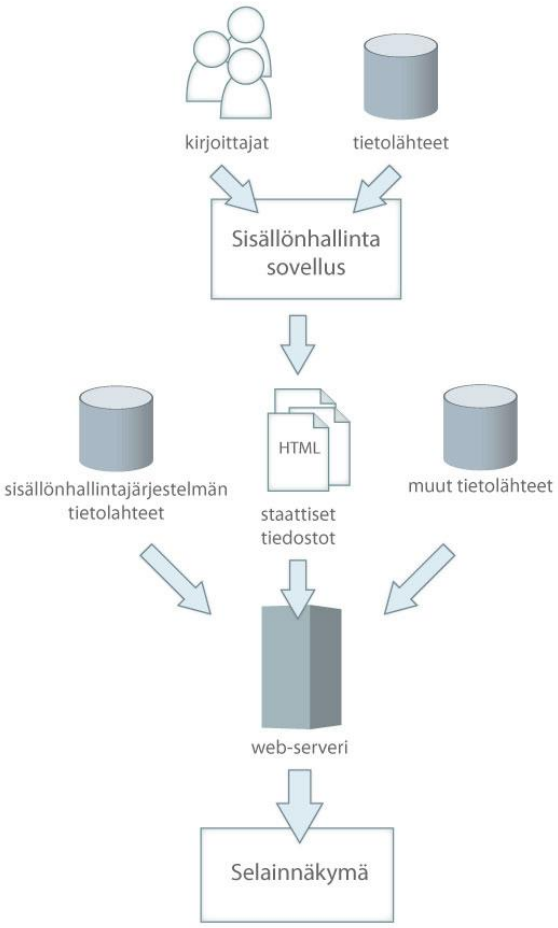
* Staattiset tiedostot: Järjestelmä sisältää joukon HTML-tiedostoja, jotka sisältävät staattisen osan järjestelmästä

* Muuttuvat tietolähteet: Näillä lähteillä tarkoitetaan järjestelmän itse luomia tietokantoja. Tietokantoja käytetään luomaan sivuille dynaamista sisältöä

* Muut tietolähteet: Web-sivuille saattaa kuulua muitakin tietokantoja, jotka eivät ole suoraa yhteydessä sisällönhallintajärjestelmään. Tätä toimintaperiaatetta tarvitaan mm. jos sivut tarvitsevat dynaamista sisältöä, jota ei muuteta järjestelmän kautta.

* Sivupohjat: Sivupohjat yhdistävät sivujen staattisen ja dynaamisen sisällön ja luovat näistä valmiin web -julkaisun. Sivupohjilla hallitaan web -sivun ulkoasuun liittyviä ominaisuuksia.

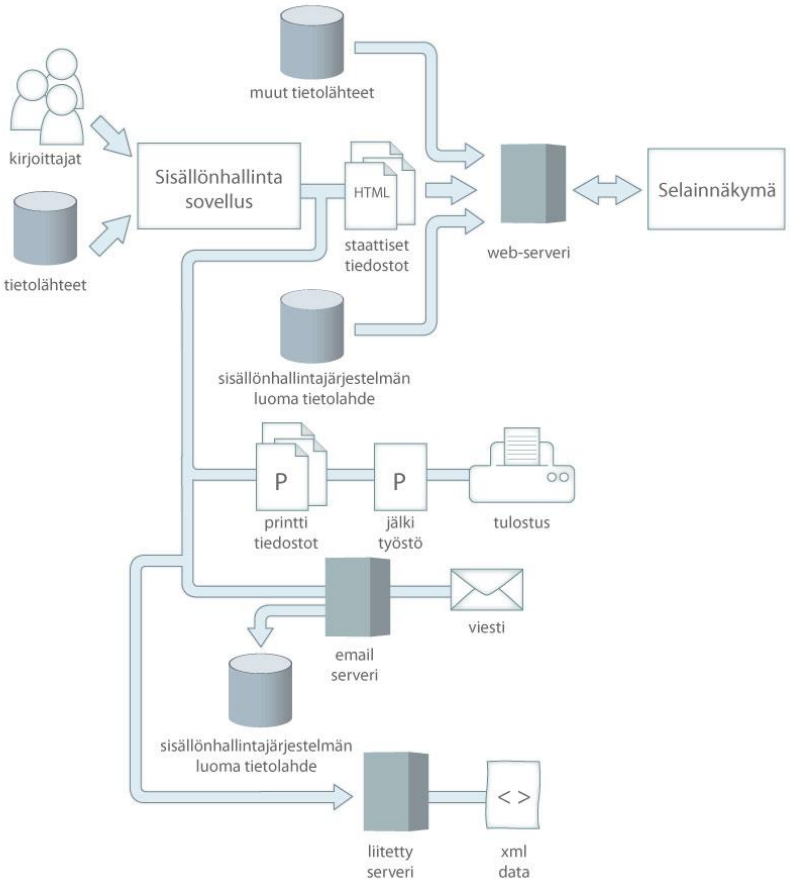
Tyypillisesti sisällönhallinnan kanssa toimiva web-sivu sisältää suurimman osan näistä osioista ja piirteistä. Kuva 3. on mukaelma Boikon kaaviosta, jossa esitetään web-sivun toiminta, joka sisältää kaikki nämä ominaisuudet. Suunniteltaessa omaa tai valitessa sopivinta järjestelmää on hyvä aloittaa tutkinta listaamalla tarvittavat ominaisuudet ja tämän jälkeen katsoa, mihin ne sijoittuvat järjestelmän rakenteellisessa puussa.(Boiko, 2005, 77-79.)



Kuva3. Mukaelma Boikon kaavio sisällönhallintajärjestelmän sisältävästä www-sivusta

2.5.5 Täysimittainen www-sisällönhallintajärjestelmä

Sisällönhallintajärjestelmiä käytetään valtaosaksi web-sivujen hallinnointiin. Pidemmälle kehitetty järjestelmä voi kuitenkin hoitaa organisaation tiedonkulussa tämän lisäksi huomattavan määrän muita tehtäviä. Boiko esittää täydellisen sisällönhallintajärjestelmän systeeminä, jonka avulla organisaatio kykenee huolehtimaan kaikista julkaisuista kaikissa medioissa saman työkulun kautta. Kuva 4. on mukaelma Boikon esimerkistä, joka esittelee täydellisen sisällönhallintajärjestelmän ominaisuuksia.



Kuva4. Mukaelma Boikon kaavio täysimittaisesta www-sisällönhallintajärjestelmästä

Täydellinen sisällönhallintajärjestelmä on suurille organisaatiolle ideaalinen keino vähentää työkustannuksia ja tehostaa julkaisuprosesseja. Täydellinen sisällönhallintajärjestelmä ei ole varsinaisesti www-sisällönhallintajärjestelmä, mutta se sisältää kuitenkin kaikki samat ominaisuudet. On myös tyypillistä, että www-sisällönhallintajärjestelmä kehitetään ja uusien päivitysten myötä järjestelmä saa ominaisuudet uusien medioiden sisällönhallintaan. Laajemman mittakaavan järjestelmän avulla voidaan hallinnoida myös ulkopuolisia sivustoja erillisten serverien kautta.

2.6 Www-sisällönhallintajärjestelmillä tavoiteltavat hyödyt

Www-sisällönhallintajärjestelmillä tavoitellaan samoja etuja kuin monilla muillakin sisällönhallintajärjestelmillä. Sisällönhallinnan yleisimpiin ongelmiin kuuluu piirre, että sisältöä tuotetaan useissa eri osastoissa, jotka eivät välttämättä ole suo-

raa tekemisissä toistensa kanssa. Informaatiovirrassa ilmenee pullonkaula julkaisuprosessin aikana ja tämä vaikuttaa suoraa julkaisujen kustannustehokkuuteen. Pullonkaula sisällönhallinnassa tapahtuu tyypillisesti ylläpitäjän kohdalla, jolla on suurimmat käyttöoikeuden ja suurin vastuu sisällönhallintatehtävissä. Pahimmassa tapauksessa sivuilla tapahtuvien päivitysten on kaikissa tilanteissa kuljettava tämän ylläpitäjän kautta. Osastot tai sisältöä tuottavat henkilöt saattavat käyttää sisällöntuottamisessa ja julkaisussa poikkeavia menetelmiä jolloin julkaisuissa ilmenee epäyhtenäisyyksiä. Hyvä ja organisaatiolle sopiva järjestelmä tuo mahdollisuuden hallita informaatiota tehokkaasti ja yhtenäisesti. Järjestelmän tuomat hyödyt ovat kuitenkin suoraa sidonnaisia organisaation toimintaan ja tarpeisiin. Suurimman hyödyn järjestelmistä saavat useimmissa tapauksissa suuret organisaatiot tai yritykset, jotka käsiteltävät informaatiomäärät ovat mittavampia ja mahdolliset kasvutarpeet todennäköisempiä. (Wayne Powel, Chris Hill 2003).

Järjestelmän käytöstä haettava hyötä on hyvin tapauskohtaista, mutta tästä huolimatta voidaan mainita joitakin esimerkkejä siitä millaisia hyötyjä järjestelmiltä haetaan. Kasvavan informaation määrän myötä ylläpitäjien määrä tulee väistämättä lisääntymään. Järjestelmällä tullaan takaaman yhtenäiset päivitystoimenpiteet ja julkaisujen ulkoasu. Sivuston kattaessa uusia aihealueita ulkopuoliset tarpeet kasvattavat informaatiomäärän tarvetta ja sivulle tarvitaan laajennuksia. Sivun kehityksessä tyypillisesti uusia käsiteltäviä aihealueita ovat: monikielisyys ja bussines-to-bussines markkinoinnin kehittäminen. Sivuston kehittyessä tarve useiden eri mediatyyppien julkaisuun tulee todennäköisemmäksi. Eri mediatyyppien tulee myös olla helposti saatavilla toisiin julkaisumuotoihin, eikä ainoastaan digitaalisiin kanaviin vaan myös paikallisiin laitteisiin kuten tulostimiin. (Ashley Friedlein)

Oikein valittu sisällönhallintajärjestelmä mahdollistaa pitkälle automatisoidun työkulkuprosessin. Monimutkaisimmissa tapauksissa työkulkuun kuuluu useita yksiköitä, jotka työstävät saman julkaisun eri vaiheita itsenäisesti kuva 5. Järjestelmän avulla tiedonkulku eri yksiköiden välillä voidaan automatisoida, jolloin prosessin ensimmäisen vaiheen valmistuttua saa työkulussa seuraavana oleva ilmoituksen. (Wayne Powel, Chris Hill 2003).

Kuva 5. Powelin ja Hillin kolmivaiheisen julkaisun työkulku

Vuonna 2005 Göteborgin yliopisto teki tutkimuksen, jossa arvioitiin sisällönhallintajärjestelmän hyötyä Volvo-yhtiön käytössä. Erääksi huomattavaksi hyödyksi esiin nostettiin vaikutus brändin kehityksessä ja ylläpitämisessä. Volvon käytössä on useita kaupallisia www-sivuja ja näihin sivuihin suoraa sidonnaisia sovelluksia. Sisällönhallintajärjestelmän avulla kaikilla sivuilla on yrityksen imagoon sopiva ulkoasu ja tyyli. Sivujen designin lisäksi yhtenäisyys parantaa sivujen laatua ja toiminnallisuutta.

2.7 Järjestelmän tavoitteet pienissä organisaatioissa

Järjestelmistä hyötyvät myös pienemmät organisaatiot. Www-sivujen ylläpitotoiminnot voivat rajoittua yhden henkilön vastuulle, jolloin järjestelmän aktiivisia käyttäjiä ei ole kuin yksi. Sivut ovat kuitenkin tällaisissa tapauksissa valmistettu ulkopuolisen tekijän kautta. Ulkopuolinen tekijä vastaa, että järjestelmä ja sivut toimivat, jolloin tulevan ylläpitäjän tehtäviksi jää ainoastaan dynaamisen sisällön päivittäminen. Ylläpitäjä saa riittävät käyttöoikeudet näitä toimintoja varten jonka jälkeen sivut toimivat kokonaisuutena ilman, että ylemmillä tasoilla on ainuttakaan ylläpitäjää.

2.8 Sisällönhallintajärjestelmän tarpeen tunnistaminen

Boikon mukaan sisällönhallintajärjestelmää tarvitaan siinä vaiheessa kun ylläpidettävää sisältöä on liikaa tuotettavaksi manuaalisesti. Sisällönhallintajärjestelmistä pystytään hyödyntämään pienissäkin projekteissa ja tekemään näistä kustannustehokkaampia. Boikon ohjesääntö on toimiva perusta tunnistamaan koska organisaation tulisi ottaa käyttöön sisällönhallintajärjestelmä. (Boiko, 2005, 113)

Sisällönhallintajärjestelmän tarve voi syntyä muunkin kuin suuren julkaistavan sisällön määrän seurauksena. Yhtä suuri merkitys on myös sisällön luojien määrällä. Sisällön tullessa web-sivuille usean eri henkilön kautta tarvitaan informaation tehokkaaseen kasaamiseen lähes poikkeuksetta tarkoitukseen sopiva järjestelmä. (Boiko, 2005, 113)

Kokonaisvaltaista sääntöä tarpeiden määrittämiselle sisällönhallintajärjestelmän käyttöönottoa varten ei ole. Monilla asiantuntijoilla on omia määritelmiä mahdollisimman yleispätevän säännön määrittämiseksi ja usein nämä säännöt pohjautuvat yrityksen tavanomaiseen kehityskulkuun. Boiko on esittänyt laskukaavan, jolla voitaisiin määrittää organisaation tarve hankkia sisällönhallintajärjestelmä. Kaavassa huomioitavat asiat jaetaan neljään pääryhmään, jotka on listattu taulukossa 1.

Liikaa sisältöä • sisältöelementit • sisältötyypit	Liikaa kirjoittajia • monimutkaiset lähteet • vaihtelevat kirjoittajat
Liikaa muutosta • Sisällön tuotanto • Suunnittelun uudistus	Liikaa julkaisuja • Sisältökanavat • personointi

taulukko 1. Neljä pääryhmää (Boiko, 2005)

tekijä	kuvaus	arvo
Kirjoittajat	Kirjoittajien määrä organisaatiossa	Jos kirjoittajia on alle 20 kaavassa arvo 1. Jos kirjoittajia enenmmän arvo on määrä jaettuna 20:lla
Lähteet	Monimutkaisten lähteiden määrä	Yksi tai vähemmän arvo 1. Jos määrä on enemmän arvo on määrä jaettuna 2:lla
Komponentit	Järjestelmään kuuluvien komponenttien (sivujen, sovellusten) määrä	Määrä alle 500 arvo 0.5. Määrä alle 1000 arvo 1. Suurempi kuin 1000 arvo määrä jaettuna 1000:lla
Tyypit	Komponenttityyppien määrä	Alle 3 arvo 0.5. Alle 5 arvo 1. Suurempi kuin 5 arvo määrä jaettuna 5:llä.
Julkaisut	Erillisten julkaisujen määrä, joka tapahtuu saman järjestelmän kautta.	
Tuotanto	Luotujen, arkistoitujen tai poistettujen komponenttien määrä viikossa	Määrä alle 25 arvo on 0.5. Määrä alle 50 arvo on 1. Suurempi kuin 50 arvo on määrä jaettuna 50:llä
Personalisointi	Personalisointien aste julkaisuissa	Ilman personalisointeja arvo on 1. Arvo 2 vastaa pientä suurten segmenttien määrää. Arvo 3 vastaa suurta pienten segmenttien määrää. Jos personalisointi tapahtuu jokaisen käyttäjän kohdalla arvo on 4.
Uudelleen suunnitelu	Suurten yksikköjen (esim. sivupohja) uudistusten määrä vuodessa	Kahdelle tai vähemmälle uudistukselle arvo on 1. Jos määrä on suurempi arvo on määrä jaettuna 2:lla

Jokaisessa ryhmässä on omat aihealueensa, joiden merkitys arvioidaan erikseen ja näiden arvioiden perusteella aihealueelle annetaan laskukaavaan sopiva arvo. Taulukko 2. on tiivistelmä arvojen luomista varten ennen laksukaavaan sijoittamista.

taulukko 2sisällönhallinnan arviointi (Boiko, 2005)

Saadut arvot sijoitetaan laskukaavaan, jossa määritellyt arvot kerrotaan keskenään. Saatu tulos määrittää organisaation toimintaan sopivan tarpeen määrän käyttää sisällönhallintajärjestelmää.

Tulosten arviointi	
Tulos	Tarpeen määrä
Alle 0.25	Ei välttämätöntä tarvetta järjestelmälle
0.25-0.5	Järjestelmän käyttöönottoa voidaan harkita, jos organisaatiossa tapahtuu mahdollisesti kasvua tulevaisuudessa
0.5-0.75	Järjestelmän käyttöönottoa tulisi harkita, jos organisaatiossa tapahtuu todennäköisesti kasvua tulevaisuudessa
0.75-1	Tässä vaiheessa järjestelmän käyttöönotto ei ole vielä täysin välttämätöntä, mutta täysin harkittavaa. Tarkempien analyysien jälkeen todellinen tarve saadaan selville ja organisaation tulevaisuuden suunnitelmat on huomiota erityisen tarkkaan
1-10	Organisaation käyttöön kannattaa harkita pienen mittakaavan järjestelmää tuomaan tehokkuutta työnkulkuun
10-100	Järjestelmän käyttöönotto olisi suositeltavaa. Järjestelmä voi ollaa ominaisuuksiltaan vielä suppea, mutta laajennettavissa tarvittaessa
100-	Järjestelmän käyttöönotto on välttämätöntä.

Taulukko 3. tulosten arviointi (Boiko, 2005)

Boiko huomauttaa, että sisällönhallinta ei ole tieteenalana tarkkaa eikä käytännössä sisällönhallinnan suunnittelulle voi antaa täysin yleispäteviä teoreettisia ohjeita. Kappaleessa mainittua lähestymistapa antaa kuitenkin erinomaisen mahdollisimman objektiivisen lähestymistavan aloittaa organisaation sisällönhallinnan suunnittelu, joka usein aloitetaankin todellisten tarpeiden määrittämisellä ja tästä toteutukseen siirtymisellä. (Boiko, 2005, 113-122)

2.9 Kehitystyökalut sisällönhallinnan osana

Kehitystyökalut kuten Adobe DreamWeaver ja avoimen lähdekoodin Nvu nähdään usein yhdentyyppisinä julkaisujärjestelminä ja termeinä nämä saattavakin usein sekoittua. Kehitystyökalut ovat sovelluksia, jotka sisältävät samoja ominaisuuksia kuin useimmat sisällönhallintajärjestelmät. Nämä sovellukset kuitenkin keskittyvät stabiilimpien ominaisuuksien hallintaan, eikä niitä käyttämällä ole mahdollisuutta luoda yhtä rikasta ja dynaamista sisältöä kuin sisällönhallintajärjestelmillä. Nimensä mukaisesti kehitystyökalut ovat tarkoitettu sovellusten kehittämistä varten. Web-kehitystyökaluilla työskennellään web-teknologiaympäristöissä, jotka pohjautuvat erinäisiin ohjelmointikieliin. Tyypilli-

simmät ohjelmointikielet, joita web-kehitystyökalut tukevat ovat; HTML, CSS, PHP, XML ja Javascript. Koodieditoriominaisuuksien lisäksi monet kehitystyökalut sisältävät myös muita laajennuksia kuten; projektihallintajärjestelmän, automaattisen koodin täydennyksen ja tuet sekä FTP- ja ja SSH-protokollille.

2.10 WYSIWYG-kehitystyökalut

Tällä hetkellä lähes kaikki kehitystyökalut sisältävät myös WYSIWYG (eng. lyh. What you see is what you get) -tyylisen sivumuokkaimen. Termillä tarkoitetaan mahdollisuutta muokata sisältöä ja näyttää se muokkaustilassa hyvin samantyylisesti kuin itse lopputuloksessa. Metadataa tai lähdekoodia ei tällöin näy muokkaustilassa laisinkaan. Web-sovelluksiin tarkoitettuja WYSIWYG-ohjelmia kutsutaan usein myös nimellä graafiset web editorit. Käyttäjä ei pysty käsittelemään suoraa kaikkea sivun sisältöä kuten metadataa eikä hänellä tarvitse olla osaamista sivun toiminnallisista ominaisuuksista tai muotoiluasetuksista.

WYSIWYG-ohjelmilla on paljon samoja piirteitä kuin www-sisällönhallintajärjestelmillä, mutta merkittävin ero on siinä mitä osa-alueita käyttäjä voi käsitellä. Sisällönhallintajärjestelmissä käyttäjällä on pääasiallisesta mahdollisuus hallinnoida ainoastaan sivun sisältöelementtejä, kun taas kehitystyökalut antavat mahdollisuuden tehdä laajempia muutoksia, jotka yltävät sivun sivupohja (template)-tasolle. Osa kehitystyökaluista tarjoaa mahdollisuuden luoda rajoitettuja työtiloja, jossa määritellyllä käyttäjällä on oikeus muokata ainoastaan tiettyjä elementtejä www-sivuilla. WYSIWYG-kehitystyökalujen käyttö tehokkaana ja monipuolisena ylläpitotyövälineenä kuitenkaan useassa tapauksessa ei ole kovinkaan käytännöllistä monissa ohjelmissa esiintyvien samojen ongelmien takia. WYSIWYG-kehitystyökalut tuottavat heikosti optimoitua lähdekoodia eivätkä kehitystyökaluilla tuotetut sivut välttämättä toimi odotetulla tavalla kaikilla selaimilla ja selainasetuksilla..

3. JÄRJESTELMÄN VALINTA

3.1 Oman järjestelmän kehittäminen vai valmis järjestelmä

Oman järjestelmän kehityksen hyödyt ja haitat Merkittävä osa valintaprosessia on lähtökohdan päättäminen rakennetaanko sivut valmiin järjestelmän ympärille vai kehitetäänkö tähän tarkoitukseen oma järjestelmä. Suurissa organisaatioissa näiden valintojen tekeminen on erityisen tärkeää ja väärillä valonnoilla voi olla huomattavia taloudellisia menetyksiä.. Friedlein aloittaa sisällönhallintajärjestelmän valinnan punnitsemalla näitä kahta vaihtoehtoa.

Oman järjestelmän kehittäminen on tyypillistä suurien organisaatioiden yhteydessä. Järjestelmän rakentaminen vaatii osaavan kehitystiimin, jota tarvitaan myös järjestelmän valmistumisen jälkeenkin. Järjestelmän kehittäminen on pitkä aikainen prosessi huolimatta siitä kuinka tarkkaan vaadittavat ominaisuudet on kartoitettu. Kehityksessä on otettava huomioon myös pitkäaikaisemmat suunnitelmat. Vastaako järjestelmä organisaation tarpeita pitkällä aikavälillä. (Friedlein, 2003, 74)

Kehitystyön hyödyt tulevat ainoastaan esiin silloin jos järjestelmä pystyy kilpailemaan muiden markkinoilla olevien tuotteiden kanssa. Pitkäkestoisen kehitysprosessin takia järjestelmä joutuu myös kilpailemaan nopeasti kehittyvän alan kanssa. Uusia sisällönhallintajärjestelmiä kehitetään jatkuvasti ja organisaation tarpeita vastaava järjestelmä saattaa ilmestyä markkinoille kesken kehitysprosessin. Lähes poikkeuksetta valmiit kaupalliset järjestelmät ovat myös organisaatiolle edullisempi vaihtoehto kuin oman järjestelmän kehittäminen. (Friedlein, 2003, 75)

Juuri omia tarpeita vastaavat ominaisuudet ovat itse kehitetyn järjestelmän suurin hyöty. Monien tulkintojen mukaan yksilöidyn järjestelmän hyödytään suurissa kokoonpanoissa, mutta usein käytännössä itse kehitettyjä järjestelmiä käytetään hyvinkin pieniin web-sivuihin-, sovelluksiin. Mm. pienet mainosmaiset sivut eivät tarvitse monitoimista järjestelmää toimiakseen vaan kaiken tarvittavan saa hoidettua yhden pienen ylläpitosovelluksen avulla. Valmiilla järjestelmillä voidaan hoi-

taa tälläisten sivujen ylläpitotoiminta, mutta samalla järjestelmä tarjoaa ominaisuuksia, joita ylläpitäjä ei tarvitse nyt tai jatkossakaan.

3.2 Valmiin järjestelmän edut ja haitat

Valmista järjestelmää ostettaessa ominaisuuksien sovittaminen tavoitteisiin on yksi suurimmista haasteista. Valmiin järjestelmän, joka vastaa juuri organisaation tarpeita voi olla vaikeaa. Usein organisaatio voi myös joutua hankkimaan itselleen järjestelmän, joka tarjoaa enemmän ominaisuuksia kuin olisi tarvetta ja tästä seuraa väistämättä ylimääräisiä kustannuksia. Ulkopuolisen toimittajan kautta hankittu järjestelmä tarkoittaa myös sitä, että tulevaisuudessa tehtävät päivitykset ja huoltotoimenpiteet on tehtävä ulkopuolisen palvelutarjoajan kautta. Tällöin organisaation vastuu järjestelmän kehittämisestä väistyy, mutta ulkopuolisen toimittajan palveluihin ei voi koskaan luottaa täysin takuusopimuksista huolimatta. Järjestelmän kilpailukykyisyys nopeasti muuttuvilla markkinoilla jää myös täysin ulkopuolisten tahojen vastuulle. (Frieldlein 75, 2003)

Varsinkin keskisuurten ja pienten organisaatioiden suosiossa ovat valmiit ilmaiset sisällönhallintajärjestelmät, joidenka markkinaosuus kasvaa kiihtyvällä tahdilla. Ilmaiset järjestelmät ovat myös usein avoimen lähdekoodin sovelluksia ja näitä järjestelmiä käytetäänkin usein pohjana räätälöidylle järjestelmälle. Avoimen lähdekoodin järjestelmät mahdollistavat tällä tavoin myös suurille yrityksille toimivan budjettiratkaisun järjestelmän hankintaa varten.

3.3 Kaupalliset järjestelmät

Hankkiessa kaupallista järjestelmää yleisesti ostaja ei ole etsimässä itselleen vain valmista sisällönhallinjärjestelmää vaan hakee hyvää kumppania verkkopalveluiden ja julkaisujärjestelmän ylläpitoon. Tässä on merkittävin ero vertaillessa kaupallisia ja ilmaisia avoimen lähdekoodin järjestelmiä. Yritykset, jotka kauppaavat omaa järjestelmää tarjoavat myös samalla palveluita, joilla he takaavat järjestelmän toimivuuden myös tulevaisuudessa. Avoimen lähdekoodin järjestelmien kehitys on täysin riippuvainen yhteisöllisien kehittäjien toimintaan, jolloin järjestelmän käytön aloittaneella yrityksellä on oltava oma it-osasto vastaamassa järjestelmän ylläpidosta. (Perttu Tolvanen, 2008, www.vierityspalkki.fi)

Avoimen lähdekoodin järjestelmät kehittyvät nopeasti ja valtaavat sisällönhallintajärjestelmien markkinaosuutta kasvavalla tahdilla. On kuitenkin huomioitava, että määrä ei korvaa aina laatua eikä varsinkaan tuen saatavuutta. Tämän takia monet kallistuvat valintaprosessissa kaupallisiin tuotteisiin. Monilla ohjelmistoyrityksillä on oma järjestelmä, jonka toiminta on keskittynyt tietyn tyylisiin palveluihin. Kaupallisen järjestelmän hankinta onkin kannattavinta jos organisaatiolla on tarve saada käyttöön järjestelmä, joka luo jonkin tasoista kilpailuetua. Juuri oikean järjestelmän löytäminen on kuitenkin haastavaa, sillä pelkästään suomen markkinoilla liikkuu satoja järjestelmiä. (Perttu Tolvanen, 2008, www.vierityspalkki.fi)

Kaupallisiin järjestelmiin vaikuttaa merkittävästi sisällönhallinjärjestelmien suhteellisen nuori ikä. Sisällönhallintajärjestelmien kehittäminen on päässyt hyvää vauhtiin vasta muutaman viime vuoden aikana, joten muutoksia järjestelmien kehitys-, myynti-, ja markkinointityylissä on lähes varmasti tiedossa. Tyypillinen ilmiö järjestelmien kehityksessä tällä hetkellä on kaupallisten ja suljettujen lähdekoodien järjestelmien siirtyminen avoimeen lähdekoodiin. Toinen vallitseva ilmiö on avoimen lähdekoodin järjestelmiin lisätyt kaupalliset lisenssit. Huolimatta siitä että järjestelmää tarjotaan avoimella lähdekoodilla, voidaan samasta järjestelmästä myös kaupata lisensoitua versiota ja maksullista ammattitukea järjestelmän kehitystoimintoja varten. Yhtenä esimerkki järjestelmänä tästä ilmiöstä voidaan mainita eZ Publish. eZ Publish on avoimen lähdekoodin järjestelmä, jonka kehityksestä vastaa pääasiassa yhteisön sijaan järjestelmän toimittanut Norjalainen emoyhtiö. eZ Publishin voi ladata maksutta ja siitä on tarjolla myös useita maksullisia lisenssejä joihin tarjotaan kaupallista tukea. Lisenssien avulla avoimen lähdekoodin järjestelmää voidaan kaupata kattavana ja luotettavana vaihtoehtona. eZ Publish on kuitenkin saannut runsaasti kritiikkiä osakseen sekavan lisenssointinsa takia. (eZ Publish 2009)

3.4 Ilmaiset järjestelmät ja avoimen lähdekoodin ratkaisut

Ilmaiset sisällönhallintajärjestelmät ovat kasvattaneet suosiotaan lähivuosina merkittävästä ja niiden kehitys ei ole koskaan ollut aktiivisempaa. Tyypillinen ilmai-

nen järjestelmä on toteutettu avoimella lähdekoodilla, joka mahdollistaa mittavan yhteisöllisien joukon ohjelmistokehittäjiä.

3.5 Käytetyimmät avoimen lähdekoodin järjestelmät

Digitaalinen mediatoimisto Water and Stone teki vuonna 2008 tutkimuksen avoimen lähdekoodin web-sisällönhallintajärjestelmien eri markkinaosuuksista. Tutkimuksessa järjestelmien käyttömäärä mitattiin arvioimalla järjestelmän hyväksynnän määrää ja brändin vahvuutta. Hyväksynnän määrä laskettiin järjestelmän latausten, asennusten ja kolmannen osapuolen tuen määrällä. Brändin vahvuus määriteltiin hakukonenäkyvyyden, suosion, julkisen media näkyvyyden sekä yleisen arvostuksen perusteella. Tällä tavoin Water And Stonen tekemässä tutkimuksessa pyrittiin saamaan mahdollisimman realistinen kuvaus järjestelmien tämän hetkisestä markkinatilanteesta. Tutkimukseen otettiin mukaan 19 järjestelmää, joiden käyttö oli tutkimuksen tekohetkellä selvästi laajamittaisimmin käytössä. Dokumentoimalla järjestelmien todellista käyttäjämäärää pyrittiin saamaan tärkeää informaatiota, jota voidaan hyödyntää mm. järjestelmän valintaprosessissa. (Ric Shreves, 2008)

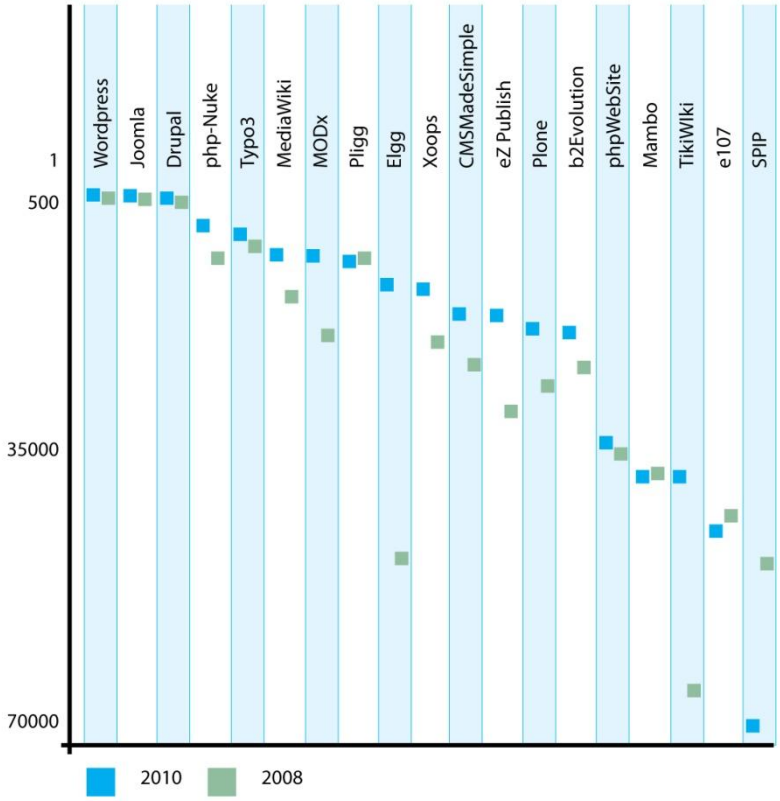
Tutkimuksessa tilastotietoja otettiin huomattavia määriä google.com:n ja Alexa.com:n palveluiden kautta. Taulukko 5:een on listattu joitakin tutkimuksessa saatuja tuloksia. Tutkimalla tämän hetkisiä googlen ja Alexan tilastoja voidaan huomata, että Water and Stonen tekemän tutkimuksen tulokset ovat kelvollisia vielä tälläkin hetkellä. Tilastotuloksissa on tapahtunut joitakin muutoksia, mutta suhteellinen markkinaosuus on pysynyt lähes samana järjestelmien kesken. Listassa mainitut yritykset Elance ja Guru ovat freelance kehittäjiä palkkaavia toimistoja.

	Keskimääräinen latausten määrä viikossa	Julkaistujen kirjojen määrä	Kehitystyöntekijöitä / asiantuntijoita

			Elance	Guru
Joomla	37762	25	2281	785
Wordpress	146847	11	1844	495
Drupal	tietoa ei saatavilla	12	933	353
Typo3	tietoa ei saatavilla	7	71	34
php-Nuke	tietoa ei saatavilla	4	47	70
Xoops	1683	3	43	27
MODx	2626	0	41	12
MediaWiki	tietoa ei saatavilla	2	38	26
Mambo	2816	3	24	117
Plone	tietoa ei saatavilla	8	32	34
Pligg	620	0	31	7
e107	4044	1	18	10
b2evolution	1006	0	14	5
TikiWiki	1056	0	9	11
phpWebSite	56	0	9	4
eZ Publish	tietoa ei saatavilla	2	6	4
CMSMadeSimple	tietoa ei saatavilla	0	6	3
Elgg	7	1	4	1
SPIP	0	0	1	3

Taulukko 5. Water and Stonen tutkimuksen tuloksia

Vertailemalla Alexa.comin eri vuosien tilastoja järjestelmien kotisivujen verkkoliikenteestä voidaan todeta, että Ric Shrevesin listaamien 19 järjestelmän markkinaosuus on pysynyt lähestulkoon samana. Samoista tilastoista kuitenkin tulee myös ilmi järjestelmien kotisivujen lähes poikkeuksellinen käyttäjämäärien kasvu myös vähemmän suosittujen järjestelmien osalla. Nämä tilastot osoittavat www-sisällönhallintajärjestelmien yleistymistä. Suosiota erityisesti on kasvattanut avoimen lähdekoodin järjestelmät sillä tutkiessa samoja tilastotietoja kaupallisista järjestelmistä sivustojen käyttäjämäärien kasvu on lähes pysähtynyttä.



kuva 6 Alexa.com tilasto järjestelmien kotisivujen liikenteestä vuosina 2008 ja 2010. Y-akselin luku vastaa sivuston sijoitusta kaikkein käytetyimpien web-sivujen joukossa.

	WordPress	Jooma	Drupal	php-Nuke	Typo3	MediaWiki	MODx	Pligg	Elgg	Xoops
2008	645	845	1223	8436	6861	13290	18339	8436	46931	19159
2010	290	423	717	4229	5340	7883	8137	8803	11804	12395

	CMSMadeSimple	eZ Publish	Plone	b2Evolution	phpWebSite	Mambo	TikiWiki	e107	SPIP
2008	22068	28027	24801	22408	33534	36014	63873	41397	47511
2010	15624	15714	17473	17920	32087	36347	36376	43422	68392

kuva 7 Edellisen kaavion tulokset tarkkoina arvoina.

3.6 Yleisiä avoimen lähdekoodin järjestelmiä

3.6.1 TYPO3

TYPO3 on yksi yleisimmin käytetyistä keskisuurista avoimeen lähdekoodiin perustuva www-sivujen sisällönhallintajärjestelmä, jonka kehittäminen alkoi vuonna 1997 tanskalaisen Kasper Skårhøj toimesta. TYPO3 on toteutettu PHP -ohjelmointikielellä. Tiedon tallentamiseen käytetään MySQL-tietokantoja. GNU vapaan ohjelmistolisenssin ja aktiivisen ohjelmistopäivittämisen ansiosta TYPO3 on saavuttanut suosiota varsinkin pienten ja keskisuurten yritysten www-sisällönhallintajärjestelmänä. TYPO3:n eduiksi monissa yhteyksissä mainitaan toimivat kielituet lukuisille eri kielille, joka ominaisuutena on varsinkin pienempien järjestelmien kohdalla usein melko puutteellinen. Tällä hetkellä TYPO3 kehitys on jakautunut kahteen haaraan 4.x ja 5.x:ään. Uusia ominaisuuksia ohjelmalla luovat myös itsenäiset kehittäjät, jotka ovat vuosien aikana tuottaneet tuhansia TYPO3:een liitettäviä laajennuksia. (TYPO3, 2008)

TYPO3:n toiminta perustuu monien muiden järjestelmä tavoin sivupohjiin. Sivupohjat rakentuvat HTML-tiedostoista, joiden sisältöä ohjataan TYPO3:n omalla komentosarjakielellä Typocriptillä. Kieli ei varsinaisesti sisällä mitään toimintoja vaan kun scripti ajetaan järjestelmän läpi se muuttuu funktioiksi. Kielen syntaksi rakentuu seuraavasti: *[objektin_polku].[määre] [operaattori] [arvo]*. Järjestelmä yhdistää sivupohjat ja tietokantojen tiedot käyttäen hydyksi typoscritpiä ja luo näitä elementtejä käyttämällä sivuista selainnäkymän. (2008, TypoScript Syntax and In-Depth Study, 3)

Alla on esimerkki äärimmilleen yksinkertaistetusta typoscript-toiminnosta, ja samalla myös pelkistetystä web-sivusta, joka on toteutettu TYPO3:lla.

```
sivu = PAGE
sivu.typeNum = 0
sivu.1 = TEXT
sivu.1.value = TYPO3 CMS
```

Typoscriptissä määritellmään sisältöobjekti PAGE ja tälle objektille arvo. Kun scripti luetaan TYPO3:n läpi järjestelmä tulostaa seuraavan HTML-lähdekoodin; (2008, Modern Template Building Manual)

```
<head>
</head>
<body>
     TYPO3 CMS
</body>
```

3.6.2 Joomla

Joomla on yksi tämän hetken yksi nopeimmin kasvavista avoimen lähdekoodin sisällönhallintajärjestelmistä. Joomlan kehitys on lähtenyt liikkeelle Mambo–nimisestä ohjelmasta vuonna 2005. Joomla on toteutettu PHP:llä ja tiedon tallentamiseen se käyttää MySQL–tietokantoja. Joomla soveltuu niin suurien yritysten kuin yksittäisten henkilöiden käyttöön. Yleisimmin Joomla on kuitenkin käytössä pienten ja keskisuurten organisaatioiden sivujen sisällönhallinnassa. Joomlan toiminta perustuu sivupohjiin, tietokantoihin ja järjestelmän laajennuksiin tai sisältöyksikköihin. Sisältöyksiköt voidaan jakaa viiteen eri ryhmään; komponentteihin, moduuleihin, plugin:hin, sivupohjiin ja kielilaajennuksiin. Jokaisella ryhmällä on omat ominaispiirteensä ja toimintaan vaikuttavat ominaisuudet. (Building Web Sites with Joomla Hagen Graf, 2007, 10-30) Sisällön sijoittamista ja käyttöä sivupohjassa ohjataan suoraa sivupohjan tiedostoilla eikä TYPO3:n tapaan erillisellä komentosarjakielellä. Sivupohjaan kuuluu tyypillisesti useita eri tiedostoja. Index.php luo sivun päärakenteen, jossa määritellään komponenttien ja moduulien paikat web-sivuilla. Template.css sisältää sivujen tyylimäärittelyt. TemplateDetails.xml sisältää sivujen meta-informaation, joka liittyy itse sivupohjaan ja ylläpitotoimintoihin. Component.php on tiedosto, jolla luodaan logiikka mm. tulostusystävällisiin sivuihin. (joomla.org 2009)

3.7 Järjestelmän valintaprosessi

Oikean järjestelmän valinta on työläs ja monimutkainen prosessi, jota vaikeuttaa laaja ja noepasti kehittyvä markkina-alueena sekä lukuisa tuotetoimittajien määrä. (Powel, Gill, 2003, 43-50). Sopivimman www-sisällönhallintajärjestelmän valinta tulisi aloittaa määrittelemällä järjestelmälle asetettavat vaatimukset ja tavoitteet. Määritelmissä tulisi huomioida taloudelliset hyödy sekä järjestelmältä toivottavat ominaisuudet. Valintaprosessi tulee olla mukana mahdollisimman suuri osa organisaatiosta sillä järjestelmän vaikutus organisaatiossa ulottaa tulevia ylläpitäjiä pidemmälle. (Robertson 2002, steptwo design)

Friedlein korostaa suunnitellun valintaprosessin merkitystä valmiin järjestelmän valinnassa. Parhaimmasta valintaprosessista on lukuisia eri näkemyksiä ja näitä näkemyksiä muokkaa monet ulkopuoliset tekijät kuten; vallitsevat trendit, taloudellinen tilanne ja organisaatioiden omat toimintamallit.(Friedlein, 79, 2003). Kahdeksan osainen prosessimallin on Friedleinin näkemys mahdollisimman yleispätevästä tavasta suorittaa sopivan kaupallisen järjestelmän valinta.

1. Vaatimusten määrittäminen
2. Markkinoiden kartoittaminen;
3. Toimittajien listaaminen
4. Sopimuspyynnön laatiminen
5. Toimittajien tapaaminen ja tuote-esittely
6. Lyhennetty lista toimittajista
7. Kaupallisen sopimuksen laadinta ja palveluiden varmistaminen
8. Päätös

Tärkeintä sopivan järjestelmän valinnassa on kuitenkin tarkka markkinoiden tutkiminen ja omien tarpeiden määrittäminen. Tarjolla olevien järjestelmien perusteellisen kartoittamiseen jälkeen voidaan arvioida onko kaupallisen räätälöidyn järjestelmän hankinta kannattavampaa kuin avoimen lähdekoodin ratkaisu. Perttu Tolvanen kertoo artikkelissaan, että sisällönhallintamarkkina on vielä hyvin epäkypsä ja edellyttää ostajalta huomattavan määrän tietoa omien sisältöjen vaatimuksista ja ymmärrystä niistä ominaisuuksista mitä järjestelmältä halutaan. (Perttu Tolvanen, 2008, vierityspalkki.fi) Monet yritykset myös kauppaavat yksilöityjä ratkaisuja, jotka on rakennettu avoimen lähdekoodin ohjelman pohjalle. Nämä toteutukset antavat myös täysin oman lähestymistavan valintaprosessille, mutta

sisällyttävä samalla paljon huomioitavia asioita, kuten lisenssiehtojen oikeellisuuden.

4. SIVUJEN TOTEUTUS JOOMLASSA

4.1 CASE Sari Karhukorpi web-sivut

Opinnäytetyön toinen case-projekti on toteuttaa web-sivut Työohjaaja Sari Karhukorvelle. Sivujen toteutukseen sopivaksi järjestelmäksi valitsin Joomlan. Järjestelmän ominaisuudet sopivat hyvin pienen organisaation käyttöön, mutta antavat kuitenkin hyvän pohjan mahdollista jatkokehitystä varten. Seuraavissa kappaleissa käsitellään sivujen toteutusta Joomla sisällönhallintajärjestelmän kanssa, käyttäen tarvittaessa case-projektia esimerkkinä.

4.2 Joomlan rakenne ja FrameWork

4.2.1 Joomlan eri osat

Sivujen toteutusta varten on tärkeää tietää Joomlan rakenne ja mitä eri kokonaisuuksia järjestelmä sisältää. Seuraavissa kappaleissa perehdytään tarkemmin websivujen luontiin käyttäen Joomlaa ja tätä ennen on hyvä tietää mistä osioista järjestelmä rakentuu. Joomlan rakenne on myös hyvin yleispätevä rakennekuvaus yleisimpien sisällönhallintajärjestelmien kohdalla. Alla olevassa listassa on lyhyt kuvaus Joomlan rakenteen eri osista. (Hagen Graf, 13-15, 2007)

Front End ja Back End tilat: Useimmat web-sisällönhallintajärjestelmät jaetaan Joomlan tavoin kahteen eri tilaan. Front End- ja Back End-tilaan. Front End tila on näkymä, jonka sivujen vierailijat ja kirjautuneet käyttäjät näkevät. Back End-tila on tarkoitettu ylläpitotoimintoja varten.

Käyttöoikeudet: Sisällönhallintajärjestelmiä varten luodaan käyttäjiä ja käyttäjäryhmiä. Jokaisella käyttäjällä ja ryhmällä on heille osoitetut oikeudet sivujen yllä-

pitoa varten. Käyttäjän voidaan jakaa esim. kirjoittajiin, julkaisijoihin ja järjestelmänvalvojiin (eng. Administrator).

Sisältö: Joomlassa sisältömuotoja on useita. Yksinkertaisimmillaan sisältö voi olla vain tekstiä, mutta sisältö voi olla myös mm. kuvia, linkkejä tai mediatiedostoja. Sisällön käytön helpottamiseksi Joomlassa on mahdollisuus jakaa sisältö itse määriteltyihin ryhmiin.

Laajennukset: Komponentit, moduulit, sivupohjat ja plug-init viittaavat kaikki Joomlan laajennuksiin. Komponenteilla tarkoitetaan laajennuksia, jotka sisältävä lisäominaisuuksia ja sisältävät oman osien Joomlan ylläpito-ohjelmassa. Sivupohjat luovat sivujen visuaalisen ilmeen ja ne koostuvat vähintään yhdestä HTML-tiedostosta. Moduuleiden päätarkoitus on hakea tietoa komponenteilta ja esittää ne sivuilla sivupohjan määräämällä tavalla. Plug-in on osa ohjelmointikoodia, jolla pyritään muuttamaan järjestelmän toimintaperiaatetta halutulla tavalla.

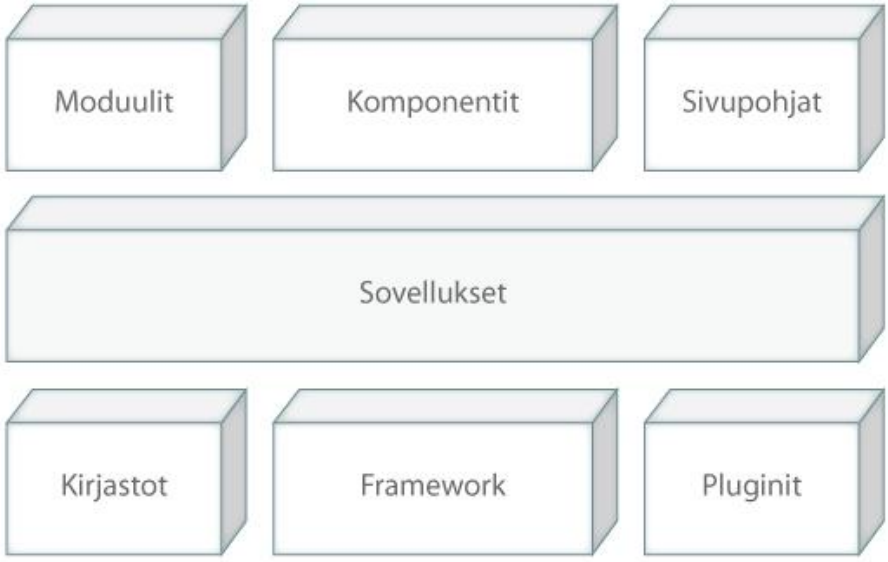
Työkulku: Työkululla käsitetään rutiineja, jotka toistuvat ylläpitotoiminnoissa. Työkulku voidaan optimoida ylläpitäjille sopivaksi ja se voidaan jakaa eri osiin jos ylläpitotehtävistä vastaa usean henkilön ryhmä.

Asetukset: Asetukset sisältävät sivuston toimintaan ja ominaisuuksiin liittyviä piirteitä, kuten sivuston avainsanat ja muut metatiedot. Asetusten kautta voidaan rajata onko sivu kaikille avoin vai vaatiiko se kirjautumisen.

4.2.2 Joomla FrameWork

Järjestelmän rakenteen osien lisäksi on hyvä tuntea sen framework eli uudelleenkäytettävä ohjelma suunnittelu. Joomlan framework jakautuu kolmeen porrastettuun tasoon, jotka on esitetty kuvassa 8. Ylimmällä tasolla ovat laajennukset, jotka vaikuttavat Joomlaan ja sen sovelluksiin. Keskimmäisellä tasolla on Joomlan sovellukset, jotka käyttävät Joomlan JApplication luokkaa. Japplication on Joomlan ohjelmoinnin perusluokka, joka tukee mm. useita ohjelmointirajapinnan (eng. lyh. API) funktioita. Tällä hetkellä Joomlan asennuspaketti sisältää neljä sovellusta.

Jinstallation, joka vastaa Joomlan asentamisesta ja poistamisesta palvelimelta. Jadministrator vastaa back-end järjestelmävalvonnasta. Sivun front-end näkymästä vastaa Jsite. Viimeinen sovellus XML-RPC tukee Joomla sivun etäylläpitoa. Viimeinen kerros koostuu kolmesta osasta. Yksi näistä osista on varsinainen Joomlan Framework siihen kuuluvine ohjelmointiluokkineen. Alimpaan tasoon kuuluu kirjasto funktioista FrameWorkin ja laajennusten käyttöön. Viimeisenä osana alimpaan tasoon kuuluvat pluginit, jotka laajentavat frameworkin toimintaa. (Joomla Documentation, 2009)



Kuva 8. Joomla framework

4.3 Joomlan asennus

Joomlasta on saatavilla yksinkertaisia asennuspaketteja, mutta ennen asennusta on sille valmistettava sopiva asennusympäristö, joka täyttää järjestelmän vaatimukset. Ensimmäisenä lähtökohtana on sopiva serveri-ympäristö. Joomla vaati toimiakseen Apache:n, joka on versio 1.13.19 tai uudempi tai Microsoft IIS:n. Palvelimen täytyy tukea PHP 4.3 tai uudempaa ja tukea MySQL-tietokantoja ja Zlib-ohjelmakirjastoa. MySQL:n versio tulee olla 3.23.x tai uudempi ja Unicode merkistöllä vähintään 4.1.x. CASE-projektissa tähän tarkoitukseen ostettiin järjestelmän vaatimukset kattava web-hotellipalvelu shellit.org:sta. Pavelun tarjoama web-

sivun liikenneraja (10Gt/kk) riittää pienen organisaation tarpeisiin. Web-hotelliin sisältyy 300Mt levytilaa ja yksi tietokanta, joka riittää web-sivujen toteutukseen, koska sivujen kaikki ominaisuudet tullaan toteuttamaan Joomlassa. (Hagen Graf, 28, 2007)

Asennuksen yhteydessä Joomla asentaa palvelimelle vaadittavat tietokannat, jotka päivittyvät järjestelmän käytön myötä. Valmiissa asennuksessa on myös mukana vähintään yksi valmis sivupohja ja siihen kuuluvat tiedostot. Uuden sivun tekeminen voidaan aloittaa valmiin pohjan päälle tai luomalla täysin uusi tyhjä sivupohja. Joomla asentaa myös joukon oletus laajennuksia, joilla on omat tiedostonsa. Joomla 1.5 versio ja uudemmat sisältävät myös valmiiksi asennetun javascript-kirjaston Mootools 1.11. (joomlaportal.org, 2007). Mootools on Joomlan toiminnan kannalta välttämätön ja tämä on hyvä huomioida jo heti sivuston suunnittelun alkuvaiheessa. Mootools ei estä muiden javascript-kirjastojen käyttöä, mutta se aiheuttaa kuitenkin ylimääräisiä toimenpiteitä. Kuvassa 9. on esitetty näkymä Joomlan asennuksen mukana tulevista tiedostoista ja kansioista sekä tietokannoista havainnollistamaan mitä asennuksen aikana siirtyy palvelimelle.

	Taulu	Toiminnot	Rivit [1]	Tyyppi	Aakkosjärjestys	Koko	Ylijäämä
	jos_banner		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_bannerclient		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_bannertrack		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_categories		2	MyISAM	utf8_general_ci	5,1 kt	-
	jos_components		32	MyISAM	utf8_general_ci	7,4 kt	-
	jos_contact_details		1	MyISAM	utf8_general_ci	3,5 kt	-
	jos_content		3	MyISAM	utf8_general_ci	28,4 kt	17,5 kt
	jos_content_frontpage		1	MyISAM	utf8_general_ci	2,0 kt	9 tavua
	jos_content_rating		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_core_acl_aro		2	MyISAM	utf8_general_ci	6,1 kt	-
	jos_core_acl_aro_groups		11	MyISAM	utf8_general_ci	4,5 kt	-
	jos_core_acl_aro_map		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_core_acl_aro_sections		1	MyISAM	utf8_general_ci	6,0 kt	-
	jos_core_acl_groups_aro_map		2	MyISAM	utf8_general_ci	4,0 kt	-
	jos_core_log_items		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_core_log_searches		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_groups		3	MyISAM	utf8_general_ci	2,1 kt	-
	jos_menu		5	MyISAM	utf8_general_ci	6,5 kt	-
	jos_menu_types		2	MyISAM	utf8_general_ci	3,1 kt	-
	jos_messages		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_messages_cfg		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_migration_backlinks		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_modules		17	MyISAM	utf8_general_ci	5,7 kt	252 tavua
	jos_modules_menu		3	MyISAM	utf8_general_ci	2,0 kt	9 tavua
	jos_newsfeeds		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_plugins		32	MyISAM	utf8_general_ci	6,8 kt	-
	jos_polls		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_poll_data		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_poll_date		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_poll_menu		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_sections		1	MyISAM	utf8_general_ci	3,1 kt	-
	jos_session		1	MyISAM	utf8_general_ci	27,7 kt	21,1 kt
	jos_stats_agents		0	MyISAM	utf8_general_ci	1,0 kt	-
	jos_templates_menu		2	MyISAM	utf8_general_ci	5,0 kt	-
	jos_users		2	MyISAM	utf8_general_ci	12,3 kt	-
	jos_weblinks		0	MyISAM	utf8_general_ci	1,0 kt	-
	36 taulu(a)	Summa	123	MyISAM	utf8_general_ci	158,3 kt	38,9 kt

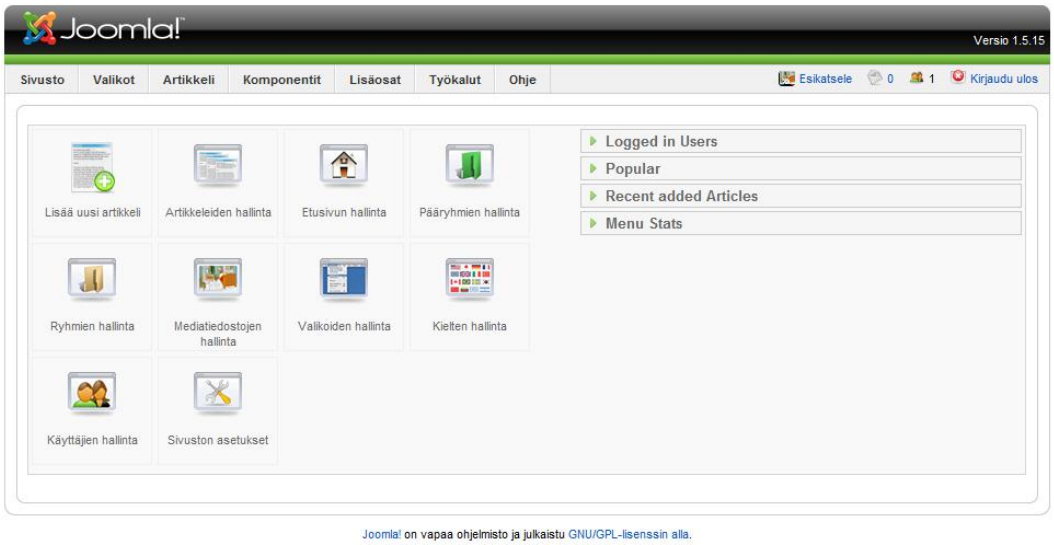
- administrator
- cache
- components
- images
- includes
- installation_
- language
- libraries
- logs
- media
- modules
- plugins
- templates
- tmp
- xmlrpc
- CHANGELOG.php
- configuration.php
- configuration.php-dist
- COPYRIGHT.php
- CREDITS.php
- htaccess.txt
- index.php
- index2.php
- INSTALL.php
- LICENSE.php
- LICENSES.php
- robots.txt

Kuvio 9. Joomlan asennuksessa tulevat tietokantojen taulukot ja asennetut tiedostot.

4.4 Joomla ylläpitäjänäkymän back-end käyttöliittymä ja sen toiminnot

Ylläpitäjänäkymän hallintapaneeli (kuva 10.) sisältää kaikki toiminnot sivuston sisällön muuttamista varten. Tätä kautta voidaan hallinnoida myös ylläpitäjätunnuksia, sivun metadataa ja asetuksia. Hallintapaneelin yläreunassa olevasta valikosta havainnollistaa kuinka sivun eri osa-alueet on jaettu ylläpitäjää varten. Sivusto osio sisältää itse sivuun liittyvät asetukset kuten käyttäjätiedot ja metadatan. Joomlassa on joukko oletustyypin valikoita, jotka löytyvät valikot valinnan alta. Sivujen perussisältömoduuleita ovat artikkelit, jotka löytyvät valikon kohdasta artikkelit. Joomlassa ei ole varsinaista sivupuurakennetta vaan sivujen sisältö koostuu sisältöelementeistä, jotka on linkitetty halutulla tavalla. Komponentit ovat Joomlan sisältöyksiköitä, joilla on omia erityisominaisuuksia: Komponentteja ovat mm. uutissyötteet ja kyselyt. Näille yksiköille on myös oma osia hallinta-

paneelissa. Hallintapaneeliin kuuluu myös Joomlan lisäosat, jotka sisältävät mahdollisuudet mm. muokata sivupohjia ja hallinnoida kielipaketteja. Tätä kautta voidaan myös asentaa uusia lisäosia. Ylläpitäjät voivat lähettää myös viestejä hallintapaneelin kautta työkalut osiossa, johon kuuluu myös välimuistiin liittyvät ominaisuudet.



Kuva 10. Joomlan ylläpitäjän käyttöliittymä

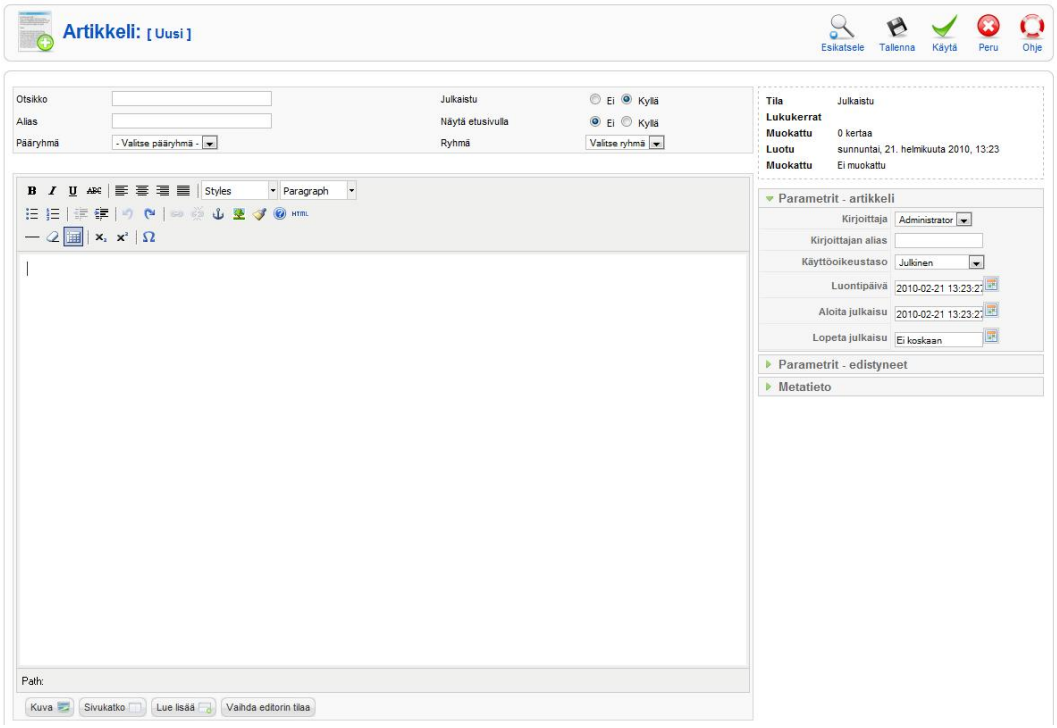
4.5 Uuden sisällön luominen

4.5.1 Uuden artikkelin lisääminen

Joomla sisältää useita eri työkaluja sisällön luomista varten ja näitä työkaluja voi myös lisätä järjestelmään erinäisten laajennusten kautta. CASE-projektin websivun sisältö koostuu lähes täysin tekstipohjaisesta sisällöstä, joten tätä projektia varten Joomlaan ei tarvitse asentaa uusia lisäosia. Tässä opinnäytetyössä Joomlan sisällön luominen esitetään muutaman esimerkin kautta, jotka liittyvät opinnäytetyön CASE-projektiin.

Yksinkertaisen tekstisisällön luonti Joomlassa onnistuu helpoiten luomalla uusia artikkeleja. Artikkelit ovat Joomlan tavanomaisimpia sisältöyksikköjä, jotka voivat sisältää tekstiä, kuvia ja linkkejä. Artikkeliin voidaan sijoittaa myös sivukat-

koksia, jotka jäsentävät artikkelin sisällön useammalle alasivulle. Varsinaisten sisällön lisäksi artikkeleilla voidaan antaa metadataa listattujen parametrien avulla, jotka vaikuttavat mm. artikkelin julkaisuaikaan. Artikkelien editori WYSIWYG-tyyppinen rich-text editori, joka sisältää tyypillisimmät tekstinkäsittelytoiminnot. Tämä editorimalli toistuu useassa muussa Joomlan sisältöeditorissa. (kuva 11.).



Kuva 11. Joomla editori artikkelin luontia varten.

Kaikki artikkelin tiedot tallentuvat Joomlan käyttämään tietokantaan content taulukkoon. Tietokannan soluja (kuva 12.) tutkiessa huomataan kuinka paljon artikkeliin tallentuu tietoa automaattisesti ja kuinka paljon artikkeli sisältää myös tietoa, joka päivittyy automaattisesti. Jokainen artikkeli sisältää mm. tiedon siitä kuinka monta kertaa tämän artikkelin selainnäkymä on ladattu.

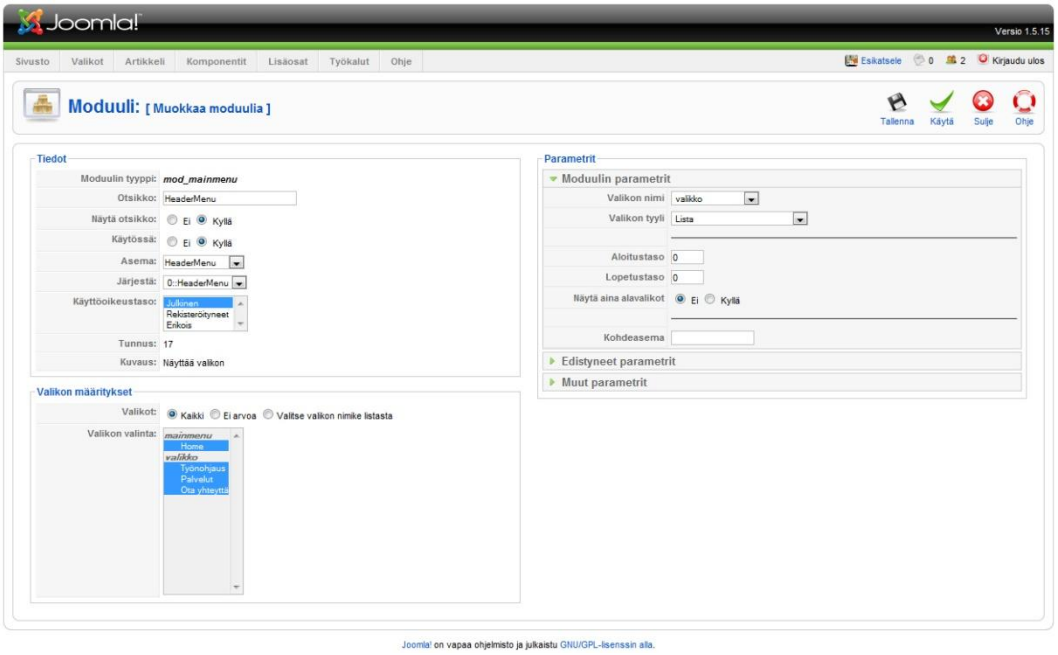
id	title	alias	title_alias	introtext	fulltext	state	sectionid	mask	catid	created	created_by	created_by_alias	modified	modified_by
1	Palvelut	palvelut		<p>Lorem ipsum dolor sit amet, consectetur adipisc...		1	1	0	1	2010-01-27 17:04:19	62		2010-02-20 17:46:27	62

checked_out	checked_out_time	publish_up	publish_down	images	urls	attribs	version	parentid	ordering	metakey	metadesc	access	hits	metadata
0	0000-00-00 00:00:00	2010-01-27 17:04:19	0000-00-00 00:00:00			show_title= link_titles= show_intro=0 show_section...	8	0	3			0	166	robots= author=

Kuva 12. Joomla artikkeli tietokannossa.

4.5.2 Valikot

Valikot ovat yksi Joomlan käytetyimmistä moduulityypeistä ja monien web-sivuprojektien tavoin CASE-projektissakin on luotava vähintään yksi valikko sivun navigointia varten. Valikot voidaan luoda moniportaisina tai yksinkertaisemmassa muodossa valikkoon valitaan tietyt sisältöelementit. CASE-projektia varten luodaan jälkimmäisellä tavalla valikko. Uusi valikko luodaan valikoiden hallinnan kautta. Valikolle tulee aluksi määrittä nimi ja otsikko. Näiden määritysten jälkeen valitaan sisältöelementit, jotka halutaan lisätä valikkoon.



Kuva 13. Joomla valikko editori.

Artikkelin tavoin myös valikoita luodessa tietokantoihin tallentuu runsaasti tietoa. Katsomalla tietokannan rakennetta (kuva 14.) saadaan myös hyvä käsitys Joomlan toimintaperiaatteista. Sivuston rakenne ei koostu yhtenäisestä sivupuurakenteesta vaan yksittäisistä sisältöyksiköistä, joiden sijoitus sivustossa on täysin riippuvainen sivun muista osista kuten valikoista.

id	menutype	name	alias	link	type	published	parent	componentid	sublevel	ordering	checked_out
1	mainmenu	Home	home	index.php?option=com_content&view=frontpage	component	1	0	20	0	1	0
2	mainmenu	Menu	menu	index.php?option=com_content&view=archive	component	-2	0	20	0	0	0
3	valikko	Työnohjaus	sisalto	index.php?option=com_content&view=article&id=3	component	1	0	20	0	3	0

checked_out_time	pollid	browserNav	access	utaccess	params	lft	rgt	home
0000-00-00 00:00:00	0	0	0	3	num_leading_articles=1 num_intro_articles=4 num_co...	0	0	1
0000-00-00 00:00:00	0	0	0	0	orderby= show_noauth= show_title= link_titles= sho...	0	0	0
0000-00-00 00:00:00	0	0	0	0	show_noauth= show_title= link_titles= show_intro= ...	0	0	0

Kuva 14. Joomla valikko tietokannoissa

4.5.3 Muut sisältöelementit

Valikoiden ja artikkelien lisäksi Joomla sisältää useita erityyppisiä sisältöelementtejä, joita ei käydä tässä opinnäytetyössä läpi. Kaksi aikaisempaa esimerkkiä kuitenkin antavat kattavan pohjan kuvamaan Joomlan sisällönluomisen toimintaperiaatteita. Oletusasennuksen lisäksi Joomlaan on saatavilla runsaasti erinäisiä laajennuksia, jotka mahdollistavat uuden tyyppisten sisältöelementtien luomisen. Tällä hetkellä Joomlan kotisivuilla on ladattavissa 4283 laajennusta. Uusien laajennusten toteuttaminen on myös tehty helpoksi Joomla Frameworkin avulla, joka Joomlan versioista 1.5 eteenpäin on suunniteltu käyttäjäystävälliseksi ja tukemaan pitkäaikaisempaa kehitystä. Aikaisemmin esitetty kolmikerroksinen mahdollistaa ohjelmistokehittäjille selkeän kehitysympäristön, jossa kehitystyö voidaan keskittää juuri oikeaan alueeseen.

4.6 Uuden sivupohjan luonti

4.6.1 Sivupohjan perusosat

Uuden sivupohjan tekeminen voidaan aloittaa muuttamalla valmiita pohjia tai luomalla täysin uudet tiedostot tätä varten. CASE-projektia varten sivupohjat luontiin täysin itse, jotta lopputulos olisi varmasti haluttu. Sivupohja pohjautuu perinteisiin HTML-, ja CSS-tiedostoihin, joten sivupohjien rakentaminen on hyvä aloittaa luomalla sivut normaalisti ilman erillisiä syntakseja Joomlaa varten. Tässä vaiheessa on kuitenkin huomioitava, että sivupohja tulee sijoittaa määrätyn hakemistorakenteen mukaisesti.

```
[joomlan hakemisto]/templates/SivupohjanNimi/
[joomlan hakemisto]/templates/SivupohjanNimi/css/
[joomlan hakemisto]/templates/SivupohjanNimi/images/
```

Määritellyn hakemistorakenteen lisäksi sivupohjan tulee sisältää tietyt tiedostot. Index.php on sivupohjassa tiedosto, joka määrittelee sivuston ulkoasun. Joomlan moduulit ovat php-pohjaisia, joten tiedoston tulee olla päätettä .php. Sivupohja tarvitsee myös erillisen XML-tiedoston templateDetails.xml määrittelemään sivupohjan metadataa. XML-tiedosto antaa tarvittavat tiedot Joomlan sivupohjan asennusta varten. Sivupohjaa asentaessa Joomlan asennusohjelma lukee PHP:llä XML-tiedoston ja poimii sieltä sivupohjan kannalta välttämättömiä tietoja kuten tiedostojen sijainnin, tiedoston kirjoittajan ja muuta oleellista metadataa. XML-tiedostoon on myös hyvä lisätä position nimisiä elementtejä. Näitä elementtejä käytetään sijoittaessa Joomlan laajennuksia sivupohjaan. TemplateDetail.xml – tiedoston rakenne on esitetty alla olevassa esimerkissä.

```
<install version="1.5" type="template">
   <name>Sari Karhukorpi</name>
   <version>0.1</version>
   <creationDate>28.07.2006</creationDate>
   <author>Kirjoittaja</author>
   <copyright>GNU </copyright>
   <authorEmail>kirjoittaja@email.com</authorEmail>
   <authorUrl>http://www.cocoate.com</authorUrl>
   <version>0.1</version>
```

```
    <description>Kuvaus sivusta</description>
    <files>
            <filename>index.php</filename>
            <filename>templateDetails.xml</filename>
            <filename>css/template.css</filename>
    </files>
    <positions>
            <position>valikko</position>
            <position>header</position>
            <position>footer</position>
    </positions>
</install>
```

4.6.2 Jdoc –lausunnot

Oleellisena osana sivupohjan tiedostoissa ovat jdoc–lausunnot, joidenka avulla sivupohjaan voidaan sisällyttää järjestelmässä luotua sisältöä. Lausunto sisältää erilaisia määreitä, joiden perusteella haetaan haluttu sisältö. Määreitä on kahta erilaista: tyyppi ja nimi. Sivuille upottaessa menu nimistä moduulia tulee index.php tiedostoon lisätä seuraava elementti.

```
<jdoc:include type=”module” name=”menu” />
```

Jdoc-lausuntoja käytetään sisällön sijoittamiseen sivuille, mutta nämä lausunnot ovat myös välttämättömiä, jotta sivun lähdekoodiin voidaan tulostaa muita Joomlan määrittämiä tietoja. Tällaisia lausuntoja tarvitaan myös joissain erikoistapauksissa. Joomlan asennus vaiheessa sivun lähdekoodissa on jdoc-lausunto <jdoc:include type=”installation” />. Tämä lausunto on käytössä ainoastaan Joomlaa asentaessa. CASE-projektia varten sivun sivupohjan lähdekoodiin välttämätön lisäys oli alla oleva lausunto.

```
<jdoc:include type=”head” />
```

Tämä lausunto on sijoittaa sivupohjan lähdekoodiin Joomlan ylläpitotilassa määriteltyjä tietoja sivuista. Lausunnon avulla sivuun linkitetään myös oikeat script-tiedostot ja metadata. Yllä oleva lausunto vastaa html-tulostuksena seuraavaa CASE-projektin tapauksessa.

```
<meta http-equiv=”content-type” content=”text/html; charset=utf-8” />
<meta name=”robots” content=”index, follow”/ >
```

Jdoc lausunto lausuntoon vaatii vähintään yhden attribuutin, mutta tarvittaessa jdoc-lausuntoon voidaan lisätä useita eri attribuutteja. Näistä usein käytetty on style-attribuutti, joka viittaa moduuliin chrome. Alla oleva esimerkki havainnollistaa useamman attribuutin jdoc-lausuntoa ja sen käyttöä.

```
<jdoc type=”module” name=”valikko” style=”rounded” id=”valikko” />
```

Yksinkertaisen sivurakenteen ansiosta CASE-projektin sivupohjaan ei tarvinnut lisätä runsaasti jdoc-lausuntoja. Sivustoon kuuluu yksi valikko, joka tarvitsi oman lausunnon. Tämän lisäksi sivuille lisättiin kaksi lausuntoa. Toinen määrittää sivuston sisällön paikan ja toinen jokaisella sivulla näkyvien yhteystietojen paikan. Alas on listattu nämä lausunnot. Jokainen lausunto pystyttiin sijoittamaan sivupohjan lähdekoodiin niille valitun -tagin sisään.

```
<jdoc:include type=”modules” name=”HeaderContent” />
<jdoc:include type=”modules” name=”headerMenu” style=”-1” />
<jdoc:include type=”component” />
```

4.6.3 Moduuli Chrome

Jdoc-lausuntoon voidaan myös vaihtoehtoisesti lisätä määre style. Tällä määreellä voidaan määrittää millaisten elementtien sisässä liitetty moduuli on lopullisessa lähdekoodissa. Käyttäessä style määrettä viitataan joomlan moduuli chromeen. Moduuli Chrome määrittelee miten moduulin tuottama sisältö tulostetaan xhtml-koodiin sivupohjassa. Moduuli chrome tuo vakiona Joomlassa tietyn määrän toimintoja, mutta niitä pystytään luomaan myös itse tarvittaessa lisää. Alla olevissa esimerkeissä havainnollistetaan style määreen toimintaa.

```
<jdoc:include type=”module” name=”menu” style=”none”/>
```

vastaava html-tulostus:

```
<ul class="menu">
  <li><!–menun sisältö --></li>
</ul>
```

```
<jdoc:include type=”module” name=”menu” style=”table”/>
```

vastaava html-tulostus:

```
<table cellpadding="0" cellspacing="0" class="moduletable_menu">
  <tr>
    <th valign="top">Main Menu</th>
  </tr>
  <tr>
    <td>
      <ul class="menu">
        <li><!–menu sisältö --></li>
      </ul>
    </td>
  </tr>
</table>
```

Yksilöidyn moduuli chromen voi sivuille lisätä luomalla sivupohjan tiedostoihin uuden tiedoston modules.php. Tämän tiedoston tulee sisältää funktio, joka nimetään seuraavasti modChrome_TYYLINIMI. Tälle funktiolle voidaan antaa kolme eri muuttujaa $module, $params, ja $attrids. Funktiossa voidaan käyttää normaaleja php-kielen määritteitä ja mooduulien ominaisuuksia. Esimerkkifunktio on yksinkertainen malli itsetuotetusta moduuli chromesta. Funktiossa on käytetty php:n if-lausetta ja moduulien ominaisuuksia.

```
<?php
   funtion modChrome_OMA($module, &$params, &£attribs){
   if(isset( §attribs[‘headerLevel’];
            §headerLevel = §attribs[‘headerLevel’];
   }else{
            §headerLevel = 3;
   }
```

```
   if (isset( §attribs[‘background’]))
            §background = §attribs[‘background’];
   } else {
            §background = ‘blue’;
   }
   echo ‘<div class=”’ .§params->get(‘moduleclass_sfx’).’”>’;
   if(§module->showtime)
   {
   echo ‘<h .§headerlevel .’>’ .§module->title .’</h’ .§headerLevel
.’>’;
   echo ‘<div class=””’ .§background .’”>’;
   echo §module->content;
   echo ‘</div>’;
?>
```

Oma moduuli chrome otetaan käyttöön jdoc-lausunnolla seuraavilla parametreillä.

```
<jdoc:include type=”modules” name=”user1” style=”OMA” />
```

Tulostus hmtl-lähdekoodissa.

```
<div>
   <h3><!—moduulin otsikko - -></h3>
   <div class=”blue”>
   <!—moduulin sisältö-->
   </div>
</div>
```

4.7 Lopputuloksen tarkastelu

Opinnäytetyön toisen CASE-projektin lopputulos vastasi alussa annettuja tavoitteita. Merkittävä osa projektin onnistumisessa oli kattavan valintaprosessin suorittaminen ennen järjestelmän valintaa. Joomla tarjosi kaikki vaadittavat ominaisuudet joita lopulliseen tuotteeseen pyrittiin luomaan ja järjestelmän käyttöön ottaminen vaati vain hyvin pienen aloituskynnyksen. Järjestelmä antaa kattavat mahdollisuudet myöhemmille päivityksille ja järjestelmän oma dokumentaatio antaa vahvan tuen tulevalle ylläpitäjälle.

5. TAPAUSKOHTAISEN JÄRJESTELMÄN SUUNNITTELU

5.1 CASE-projekti kuvagalleria valokuvaaja Teija Pekkalalle

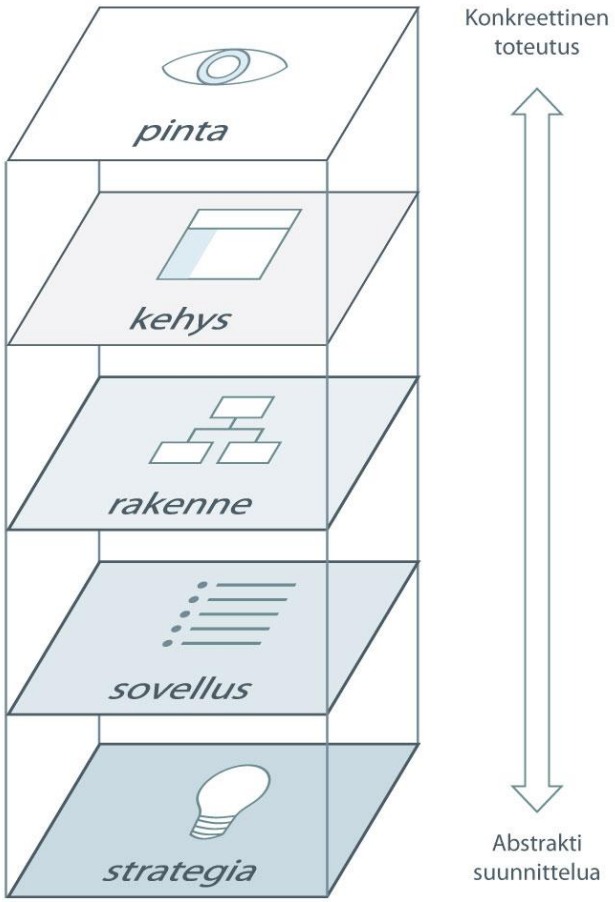
Toisessa opinnäytetyön CASE-projektissa toteutetaan www-sivut valokuvaaja Teija Pekkalalle. Sivut toimivat valokuvaajan henkilökohtaisena web-galleriana ja portfoliona. Sivut toteutetaan ilman valmista sisällönhallintajärjestelmää ja ylläpitotoiminnot tapahtuman räätälöityjen toimintojen avulla. Sivusto sisältää hyvin rajoitetun määrän toimintoja, joten oman pohjan suunnittelu sisällönhallintaa varten on toteutuksen kannalta käytännöllistä. Lisäarvoa tälle toteutustavalla antaa myös täysin yksilöllinen ja omaleimainen toteutus lopullisessa tuotteessa. Omien ylläpitotoimintojen kehittäminen aloitetaan tavoitteiden määrittämisestä ja tästä siirtymällä sovelluksen kehittämiseen käyttäen erityisesti hyödyksi Jesse Garretin luomaa mallia käyttäjä keskeisestä web-suunnittelusta. Sisällönhallintajärjestelmän suunnitteluun on lähdettävä samoista lähtökohdista kuin muidenkin web-sovellusten kehitysten kanssa.

Päätavoitteena on tuottaa helposti ja nopeasti toimivat ylläpitotoimenpiteet. Sivuston rakenne tulee pysymään aina samana, joten laajennusmahdollisuuksia uusia sivuosuuksia varten ei tarvitse huomioida. Suuri huomio keskittyy näin toimivaan järjestelmän arkkitehtuuriin ja tätä tukevaan käyttöliittymäsuunnitteluun.

Projektia lähdetään toteuttamaan php-tiedostoilla ja sivuilla päivittyvää tietoa varten käyttöön otetaan yksi MySQL-tietokanta. Toteutuksessa käytetään valmiita javascript-kirjastoa jQueryä, jonka ominaisuuksia hyödynnetään erityisesti ylläpitonäkymän käyttöliittymäsuunnittelussa. Käyttöliittymäsuunnittelun merkitys on tässä projektissa erityisen tärkeää, joten tämä osio tulee huomioida jo kuvagallerian ensimmäisissä kehitysvaiheissa. Jesse Garret käyttää käyttäjäkokemuksia tukevassa web-suunnittelussa viisitasoista mallia, jossa toteutuksen eri vaiheet on jaettu selkeästi ja johdonmukaisesti. CASE-projektin toteutuksessa web-sovelluksen kehitystyötä tarkastellaan käyttäen hyötynä Garretin luomaa mallia. Seuraavissa kappaleissa esitellään tämän mallin eri vaiheet ja esitetään kuinka niitä on käsitelty CASE-projektissa.

5.2 Suunnitteluprosessin viisi tasoa

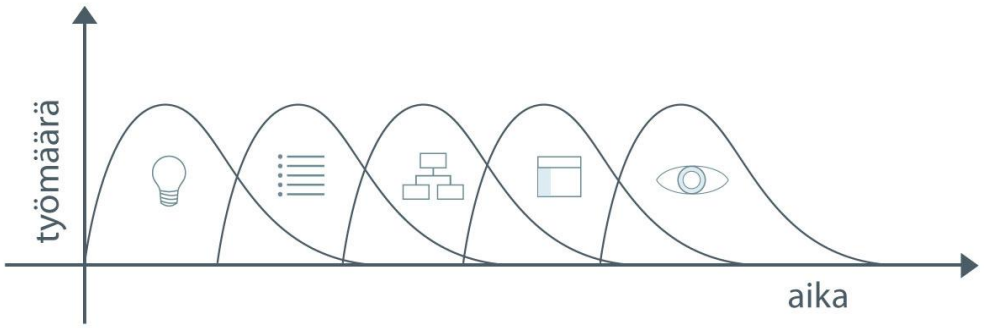
James Garret jakaa käyttäjäkokemuksiin perustuvan web-suunnittelun viiteen erilliseen päällekkäiseen tasoon. Jokainen taso käsittelee tiettyä osa-aluetta suunnitteluprosessissa. Suunniteluprosessi etenee johdonmukaisesti tasojen välillä ja tällä tavoin Garret pyrkii ottamaan huomioon kaikki web-suunnittelussa esiintyvät käytettävyysongelmat. Tasojen sisältö aloittaa sovelluksen toteutuksen käsittelemällä abstrakteja suunnittelun aspekteja ja edetessä tasoissa ylemmäksi tasojen sisältö muuttuu jatkuvasti konkreettisemmaksi. Jokainen taso on vaikuttaa suoraa sitä seuraavaan tasoon ja tällä tavoin kokonaisuutena Garretin esittämä rakenne luo toimivan rakennusmallin web-toteutuksia varten. Alimmaisena tasona on strategia taso, josta järjestyksessä seuraavat taso ovat: sovellus, rakenne, kehys ja pinta



kuva 15. (Garret, 20-30, 2002).

Strategia-taso sisältää suunnitelman sivujen sisällöstä, tarkoituksesta, käyttäjistä ja ylläpitäjistä. Strategia tason tulee sisältää kaikki sivun ominaisuudet ja tavoitteet. Sovellus-taso määrittelee mitä ominaisuuksia ja funktioita sivut tuovat käyttäjän käytettäväksi. Rakenne-taso määrittelee pääosin sivun sivupuurakenteen. Kuinka käyttäjä etenee sivuilla ja mitenkä sivujen sisältö on järjestetty. Tämän tason merkitys web-sovelluksen käyttöliittymä suunnittelussa on erityisen merkittävä. Kehikko-taso määrittelee web-sivujen tai sovelluksen visuaalisen rakenteen. Tämä taso ei sisällä varsinaisia visuaalisia elementtejä vaan rakenteen kuinka nämä elementit ovat sijoitettu sivuille. Viimeinen taso pinta sisältää sivujen tai sovelluksen lopullisen ilmeen, joka käytännössä vastaa selainnäkymää. (Garret, 20-22, 2002).

Suunniteluprosessi etenee alimmalta tasolta ylimpään ja jokaisella tasolla tehdyillä päätöksillä on suora vaikutus seuraavaan tasoon. Tämän seurauksena suunnitteluprosessia ei kannata käytännössä toteuttaa taso kerrallaan. Ensimmäisen tason ollessa lähes valmis on hyvä siirtyä jo seuraavan tason toteutukseen. Tutkittaessa seuraavan tason toteutusta huomataan edellisessä tasossa tehtyjä puutteita. Näiden puutteiden korjaaminen on vielä helppoa koska tason työstäminen ei ole saatettu vielä loppuun. (kuva 16.)



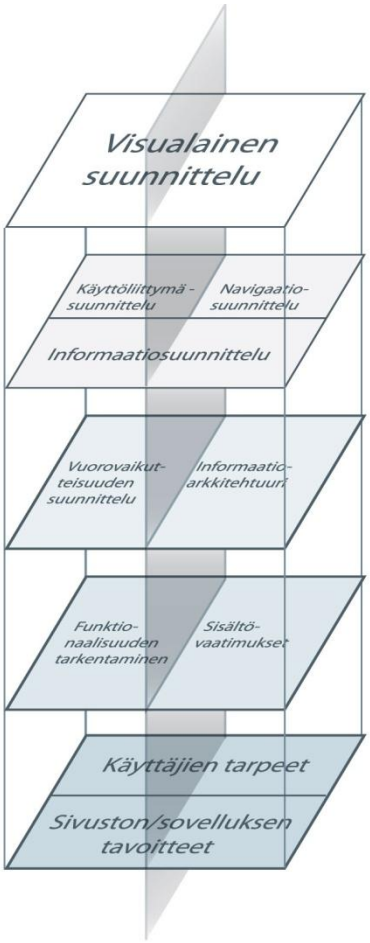
Kuva 16. Viiden tason työkulku

5.3 Tasojen tarkempi jakaminen

Sovellussuunnittelun viisi tasoa joudutaan vielä jakamaan tarkempiin pienempiin osiin, jotta malli vastaisi realistisempaa ja yleispätevää web-sovelluksen suunnittelua. Web-toteutukset ovat luonteeltaan hyvin kaksijakoisia ja tämän takia sovel-

luskehityksessä on hyvä eritellä käyttöliittymäsuunnittelu ja informaatiorakenne toisistaan. Tämä jaottelu erottaa web-sovelluksen toiminnan ja informaatiorakenteen toisistaan ja helpottaa näin suunnitteluprosessia. Jaottelu tuo myös samalla myös tarkennuksia tasojen sisältöön. Jokaisen tason kohdalla sisältö muutokset eivät ole kuitenkaan suoraa riippuvaisia sisällön jakamisesta kahteen osaan, joka on kuitenkin välttämätöntä kokonaisuuden rakentamisen kannalta. (Garret, 30-35, 2002).

Strategia-taso ei jaeta kahteen osaan edellisessä kappaleessa mainitun jaonseurauksena. Tämä taso jaetaan kahteen osaa, jotka ovat käyttäjätarpeet ja sivuston tavoitteet. Sovellus-taso jaetaan kahteen osaan, jotka erotellaan informaatiorakenteen ja käyttöliittymäsuunnittelun seurauksena. Toinen osista sisältää funktionaaliset tarkenteet web-sovelluksen toiminnoista ja toinen on sisällön vaatimukset. Rakenne taso jaetaan kahteen osaan, jotka ovat informaatioarkkitehtuuri ja interaktion suunnitteluun. Kehys-taso sisältää suunnittelussa käyttöliittymäsuunnittelun ja navigointi-suunnittelun, jotka jakautuvat edellisen kappaleen toteutuksen johdosta. Ylin taso pysyy yhtenäisenä, mutta sen sisällön merkitys voidaan tarkentaa visuaaliseen suunniteluun. Alla oleva kaavio esittää viiden tason suunnittelumallin muutokset. (Garret, 30-35, 2002).



kuva 17. Tasojen tarkempi jakaminen

5.4 Strategia-taso

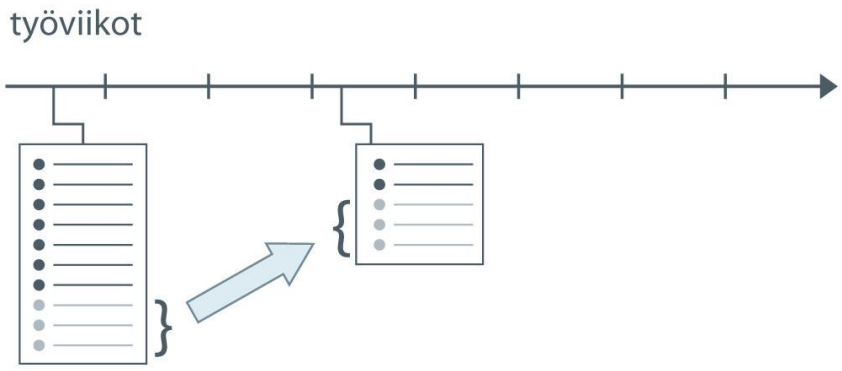
Strategiatasossa määritellään sivuston tavoitteet ja käyttäjien tarpeet. Sivuston tavoitteet voivat olla taloudellisia, keskittyä brandin luomiseen tai toiminnallisen rakenteen tuottamiseen. CASE-projektin we-sivustolla ei tavoitella taloudellista tuottoa, joten pääarvo sivuilla on luoda toimiva ja käyttäjäystävällinen kokonaisuus. Kahdesta strategia-tason osasto käyttäjien tarpeet saavat näin huomattavasti suuremman osan toteutusta suunnitellessa. (Garret, 2002, 40-50)

Suunniteltaessa käyttäjien tarpeita on prosessi aloitettava käyttäjien luokittelulla. CASE-projektissa keskitytään sivujen ylläpitäjä toimintojen kehittämiseen, joten käyttäjien luokittelu rajoittuu huomattavasti. Käyttäjät voidaan luokitella kahden

eri periaatteen mukaan. Tilastotieteeseen perustuvassa luokittelussa käyttäjät luokitellaan tietojen kuten: iän, sukupuolen, ammatin tai koulutuksen perusteella. Toinen tapa luokitella käyttäjiä on psykologinen lähestymistapa, jolloin huomioidaan käyttäjien asennoituminen ja kokemukset alueeseen, jota sivu käsittelee. CASE-projektin kannalta voidaan käyttää hyödyksi näitä molempia lähestymistapoja, koska sivujen pääylläpitäjä on jo tiedossa. Käyttäjäkartoituksen myötä voidaan suunnitella pohjaa sivun käytettävyydelle ja aloittaa käyttäjätutkimusten tekeminen. Nämä toimenpiteet ovat erittäin oleellisena osana laajamittaisissa web-projekteissa. CASE-projektin kohdalla näiden osa-alueiden suunnittelu jäi suppean käyttäjäryhmän ja sivuston yksinkertaisuuden takia hyvin vähäiseksi. Selkeän käyttäjäryhmän ansiosta järjestelmän tavoitteiden määrittäminen kuitenkin helpottui huomattavasti. Ylläpitotoimintojen tulee olla nopeat ja vaivattomat käyttää. Kuvien esitystyyliä tulee pystyä muuttamaan tarvittaessa. Kuvagalleriassa ei tarvitse olla muokkausominaisuuksia itse kuville, koska ylläpitäjällä on kokemusta kuvankäsittelystä. (Garret, 2002, 41-55)

5.5 Sovellus-taso

Scope-tasolla suunnitellaan web-projektin funktionaalisuus tarkemmin ja määritellään sisällön vaatimukset, jotka yhdessä rakentavat sivuston toiminnallisuuden. Jokaisen projektin tavoitteiden määrittäminen tulisi aloittaa listaamalla kaikki toiminnot, joita sivuille mahdollisesti kehitettäisiin ja samoin on myös hyödyllistä listata kaikki ominaisuudet joita sivuille ei tule. Tällä tavoin voidaan sivuston toimintojen kehittäminen aloittaa tehokkaasti ja projektin edetessä ylimääräisiä toimintoja voidaan karsia pois. Selkeä tavoitteiden ja toimintojen listaaminen antaa myös selkeän pohjan työprosessin aikataulusuunnitteluun. Listatut toiminnot voidaan sijoittaa projektisuunnitelman sopiviin vaiheisiin ja niiden siirtäminen on näin myös helppoa (kuva 18.). (Garret, 2002, 62-66)



kuva 18. Työtehtävien siirtoa

CASE-projektin kuvagalleria ei sisällä runsaasti erilaista sisältöä ja toiminnallisuuksia on selkeästi rajattu määrä. Alla olevassa listassa on listatuttuna sisällön vaatimukset ja tarkemmat toiminnallisuudet. Listan sisältöä voidaan verrata aiemman strategia tason sisältöön ja katsoa täyttääkö tarkemmin määritellyt toiminnallisuudet ensimmäisessä vaiheessa esitettyjä toiveita.

- Sisältö koostu kuvista ja tekstistä
- Kuvat sisältävät tarkempia tietoja
- Kuvien tietoja voidaan muokata
- Kuvien tietojen näkyminen sivuilla on ylläpitäjän päätettävissä
- Kuvagalleriassa kuvat voidaan jakaa ryhmiin
- Kuvaryhmissä kuvien järjestystä voidaan muuttaa nopeasti
- Kuvagalleriassa tulee olla vähintään yksi pääylläpitäjäkäyttäjätili

Listaa tutkimalla voidaan aloittaa järjestelmän toteutuksen suunnittelu. Tietojen tallentamiseen voitaisiin käyttää xml-tiedostoja tai tietokantoja tai näiden yhdistelmää. Järjestelmän pienimuotoisuuden takia toteutusta varten otetaan käyttöön yksi MySQL-tietokanta. Toimintoja varten käytetään php-ohjelmointia ja käyttöliittymäsuunnittelussa hyödynnetään jQuery javascript-kirjastoa, jonka avulla voidaan parantaa käytettävyyttä merkittävästi kasvattamatta kuitenkaan työmäärää kovinkaan paljon. (Garret, 2002, 75-81)

Tarkastelemalla sisällön vaatimuksia voidaan suunnitella pohja tietokannoille. CASE-projektia varten loin kolme erillistä taulukkoa. Yksi taulukko sisältää sivujen tekstisisällön, yksi taulukko gallerian kuvat ja niiden tiedot ja viimeiseen taulukkoon tallettuu ylläpitäjäsovelluksen asetukset. Tekstisisältö ja asetukset ovat taulukkorakenteeltaan yksinkertaisia tietojoukkoja. Gallerian kuvia varten on taulukko, joka koostuu kuvajoukosta. Jokaisella kuvalla on talletettuna kaikki tarvittava tieto, jota tullaan tarvitsemaan ylläpitotoiminnoissa ja sivuston toiminnassa. Tämän lisäksi taulukkoon tallettuu tietoa tulevia päivityksiä ajatellen. Jokaiselle kuvalle tallettuu vähintään seuraavat tiedot: kuvan nimi, ID-numero, kuvaryhmä, kuvateksti, kuvatiedoston nimi, esikatselukuvan tiedostonimi ja kuvan mitat pikseleinä. Näistä tiedoista suurin osa on tietoa, joka ei näy ylläpitäjälle päivitystoimintojen yhteydessä, mutta ovat toiminnan kannalta välttämättömiä.

5.6 Rakenne taso

5.6.1 Rakenne tason sisältö

Edellisillä tasoilla järjestelmän suunnittelu käsiteltiin vielä hyvin abstraktilla tasolla. Tämä tutkimus oli kuitenkin projektin kannalta välttämätöntä yhdenmukaisen suunnittelun kannalta. Edellisellä tasalla suunniteltiin sovellukselle asetut tavoitteet ja seuraavaksi joudutaan suunnittelemaan niiden rakenne ja sijainti koko toteutuksessa. Toimivan rakenteen kehittäminen sisältää kaksi erillistä osa-aluetta: vuorovaikutteisuuden suunnitteluun ja informaatio arkkitehtuurin suunnittelun. (Garret, 2002, 86-90)

5.6.2 Vuorovaikutteisuuden suunnittelu

Vuorovaikutteisuutta suunnittelemalla pyritään tutkimaan käyttäjien toimintamallia käyttäessä sovellusta. Millä tavoin käyttäjä etenee sovelluksen valikoissa ja millaisia valintoja hän tekee. Hyvän interaktiosuunnittelun tulee ymmärtää nämä toimintamallit ja ja osata vastata käyttäjän tekemiin valintoihin. (Garret, 2002, 89-90)

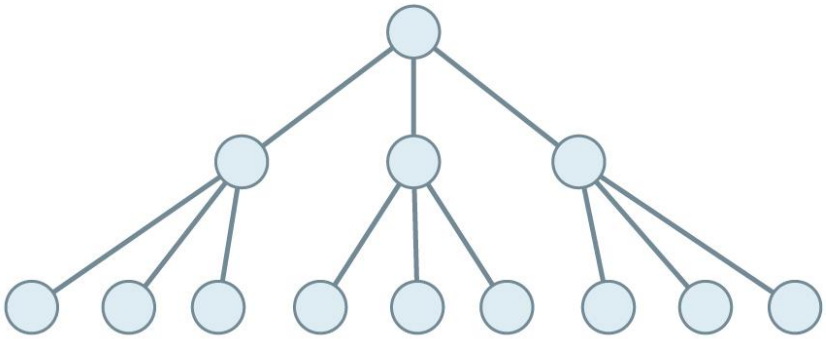
Suunnittelun tukena voidaan käyttää konseptuaalisia malleja, jotka ohjaavat käyttäjää tekemään oikeita valintoja. Hyvänä esimerkkinä näistä malleista on verkkokauppasovellusten ostoskori-toiminto. Ostoskori on käyttäjälle tuttu esine vaikka hän ei olisi koskaan käyttänyt verkkokauppaa. Käyttäjä tietää, että hän voi lisätä tuotteita ostoskoriin tai poistaa niitä. Ostosten jälkeen siirrytään maksuvaiheeseen. Konseptuaalisilla malleilla voidaan sovellukseen lisätä monimutkaisiakin ominaisuuksia, joiden käyttökynnystä vähennetään epäsuoralla ohjeistuksella. (Garret, 2002, 89-92) CASE-projektissa konseptuaalisia malleja hyödynnetään erityisesti järjestelmän käyttöliittymäsuunnittelussa. Valikoiden painikkeet suunnitellaan siten, että ne vastaavat yleisesti käytössä olevaan symboliikkaa.

Keskeinen osa toimivaa interaktiosuunnittelua on virheiden hallinta, jota ilman hyvinkin suunnittellun sovelluksen käytettävyys heikentyy huomattavasti. Lähtökohtaisesti sovellus tulee suunnitella siten, että käyttäjä ei pysty tekemään virheitä sovelluksen toimintojen yhteydessä. Virheidenhallinnan tulee noudattaa kolmevaiheista prosessia: estäminen, korjaaminen ja palauttaminen. (Garret, 2002, 92) CASE-projektissa noudetaan tätä samaa periaatetta lähes jokaisessa galleriajärjestelmän toiminnossa. Merkittävimmät toiminnot gallerian ylläpidossa ovat kuvien lisääminen, poistaminen, muokkaaminen ja järjestäminen. Tämän lisäksi järjestelmä sisältää pieniä ominaisuuksia tekstikokonaisuuksien hallintaan. Kuvien lisääminen, muokkaaminen ja siirtäminen toimivat kaikki yksinkertaisen käyttöliittymän kautta, jolla ennaltaehkäistään virheitä. Jokainen näistä toiminnoista on myös peruutettavissa saman käyttöliittymän kautta, joten mahdollisen virhetoiminnon jälkeen edellisen tilan palauttaminen on helppoa. Radikaalein muutos gallerian päivityksessä on kuvien poistaminen. Tätä varten järjestelmässä ei ole palauta toimintoa. Virheellisen valinnan voi kuitenkin korjata ennen varsinaista poistoa varmistusikkunan kautta, joka ilmestyy käyttäjälle jokaisen kuvan poiston yhteydessä. Varmista ikkunaa käytetään ainoastaan kuvan poiston yhteydessä, koska tämä vähentää huomattavasti käyttökokemuksen miellyttävyyttä eikä se ole myös näiden toimintojen kohdalla välttämätön.

5.6.3 Informaatioarkkitehtuuri

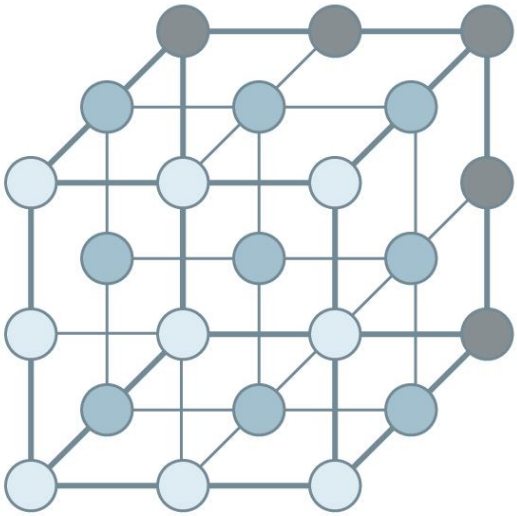
Informaatioarkkitehtuuri määrittelee sivun tai sovellukset organisoinnin ja navigointipolut. Informaatio rakenne pohjautuu useasti arkkitehtuuriseen rakenteeseen, joka koostuu informaatioarkkitehtuurin yksiköistä nodeista. Nodejen järjestys ja toisiinsa linkitys määräytyy käytetyn arkkitehtuurise nrakenteen mukaan. (Garret, 2002, 97)

Hierarkinen- tai puurakenne perustuu parent- ja child-nodeihin. Kaikilla nodeilla on parent-node ja määrittelemätön vapaa määrä child-nodeja. Periytyvä rakenne on helposti ymmärrettävä ja sitä käytetään monissa yhteyksissä, jonka seurauksena tämä rakenne malli on hyvin yleisesti käytössä. Hierarkkinen rakenne vastaa myös hyvin läheisesti tietokoneiden toimintamallia (Garret, 2002, 98) (kuva 19.)



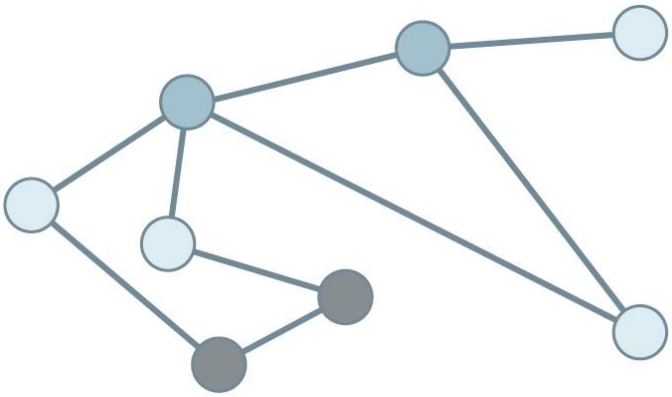
kuva 19. Hierakrinen puurakenne

Matrix-rakenne mahdollistaa moniulotteisen suunnittelumallin informaatioarkkitehtuuriin. Nodejen välinen linkitys kulkee monisuuntaisesti erityyppisten sisältöelementtien välillä. Matrix-rakennetta voidaan ajatella kolmiulotteisena rakennemallina, jossa käyttäjällä on useita mahdollisuuksia edetä sovelluksen eri osiin. Enemmän kuin kolmen ulottuvuuden rakennemalli on mahdollinen, mutta tätä käytetään hyvin harvoin, mutta tietyissä ammattilaissovelluksissa tämäkin malli on myös käytössä. (Garret, 2002, 99) (kuva 20.)



kuva 20. Matrix-rakenne

Orgaaninen rakenne on tyypillinen tapauskohtaisia ratkaisuja suunnitellessa. Tämä rakenne ei noudata mitään tiettyjä sääntöjä ja nodejen välinen linkitys koostuu kehittyvästä verkosta. Tämä rakennemalli voi toimia jatkuvasti kehittyvien sovellusten kanssa, mutta toisaalta se ei anna käyttäjälle minkäänlaista loogista rakennetta, jonka mukaan edetä sovelluksen toiminnoissa. (Garret, 2002, 99) (kuva 21.)



kuva 21. Orgaaninen rakenne

Peräkkäinen rakenne on toimintamalleista yksinkertaisin ja samalla helposti lähestyttävin, mutta ei kuitenkaan monessakaan tapauksessa käytännöllisin. Tämä malli soveltuu lähinnä ainoastaan esim. lyhyiden ohjemateriaalien käyttöliittymiin. (Garret, 2002, 100) (kuva 22.)

kuva 22. peräkkäinen rakenne

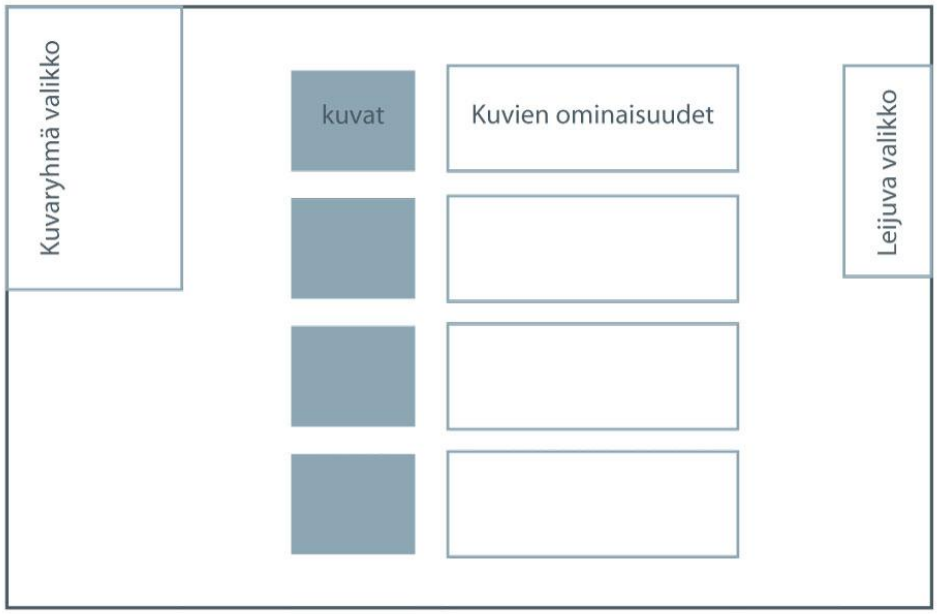
Eri informaatioarkkitehtuurirakenteiden tunteminen on erittäin hyödyllistä suunnitellessa oman sovelluksen käyttäytymistä. CASE-projektin käyttöliittymä pyrittiin toteuttamaan siten, että käyttäjällä on mahdollisuus toteuttaa samassa näkymässä lähes kaikkia ylläpitotoimintoja. Valikkojen rakenteet suunniteltiin siten, että siirtyessä järjestelmässä eteenpäin käyttöliittymätilasta ei poistu mitään toimintoja vaan lisätoiminnot ilmestyvät samaan näkymään. Jqueryllä toteutetut liukuvalikot olivat tähän tarkoitukseen käytännöllisin ratkaisu. Kuvagalleriajärjestelmän rakenne vastaa hyvin läheisesti Matrix-rakennetta. Teknisesti tämän rakenteen ominaisuuksien lisääminen tähän rakenteeseen onnistui helposti, koska galleriajärjestelmän ylläpitotoiminnot olivat jokainen oma php-tiedostonsa. Tällä tavoin järjestelmän suunnittelu nopeutui merkittävästi ja rakenteen muuttaminen ja kehittäminen oli erittäin vaivatonta.

5.7 Kehys-taso

Konseptuaalisen suunnittelun jälkeen voidaan aloittaa web,-sivun tai sovelluksen visuaalisen rakenteen suunnittelu, joka tuo ilmeen edellisillä tasoilla rakennetuille toiminnoille. Kehys-taso voidaan jakaa kolmeen pääalueeseen: käyttöliittymäsuunnittelu, navigointisuunnittelu ja informaatiosuunnittelu. (Garret, 2002, 114)

CASE-projektin osalta navigaatiosuunnittelun ja informaatiosuunnittelun merkitys ovat suhteellisen pieniä galleriasovelluksen luonteen takia, mutta käyttöliittymäsuunnittelun merkitys kehys-tason osa-alueista on erityisen merkittävä. Toimivan graafisen toteutuksen lisäksi käyttöliittymän toimintoja on muokattu joustavammiksi ja miellyttävimmiksi jQuery-kirjaston avulla. JQuery-kirjasto tarjoaa runsaasti funktioita, joidenka avulla selainpohjaisen sovelluksen käyttöliittymän ominaisuuksien kehittäminen on vaivatonta. Modal-ikkunat, liukuvalikot ja raahattavat valikot ovat kaikki toteutettu jQuery:ä käyttämällä.

Kehys-tasolla käyttöliittymäsuunnitteluun kuuluu sovelluksen rautalankamallin suunnittelu. Rautalankamalli jakaa tilan sivun tai sovelluksen eri osia varten ja määrittelee lopullisten visuaalisten elementtien sijainnin. CASE-projektin kohdalla rautalanka malli on hyvin yksinkertainen ja tyypillisestä mallista poiketen se sisältää leijuvia kappaleita, joilla pyritään parantamaan käyttäjäystävällisyyttä. Leijuvat kappaleet sisältävät valikon josta löytyy kaikki yleisimmät ylläpitotoimenpiteet.



kuva 23. Kuvagallerian rautalankamalli

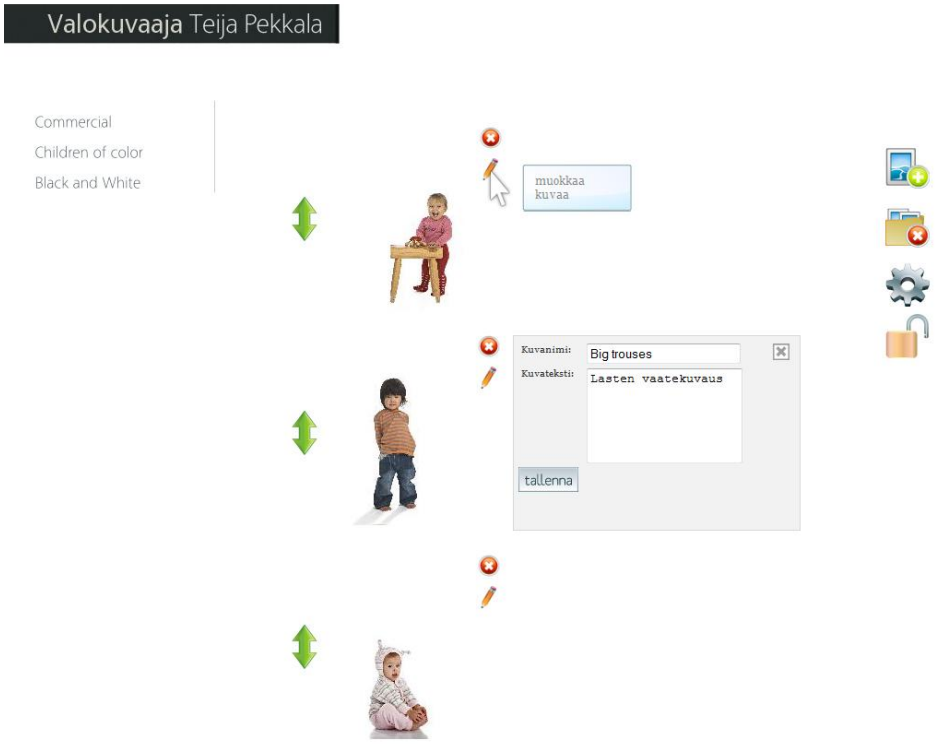
5.8 Pinta-taso

Pinta-tasoon kuuluvaa visuaalinen suunnittelu pohjautuu vahvasti edellisessä kappaleessa mainittuun rautalankamalliin. Visuaalinen suunnittelu yhdistää lopullisen toteutuksen elementtejä ja luo valmiille kokonaisuudella lopullisen ilmeensä. (Garret, 2002, 142)

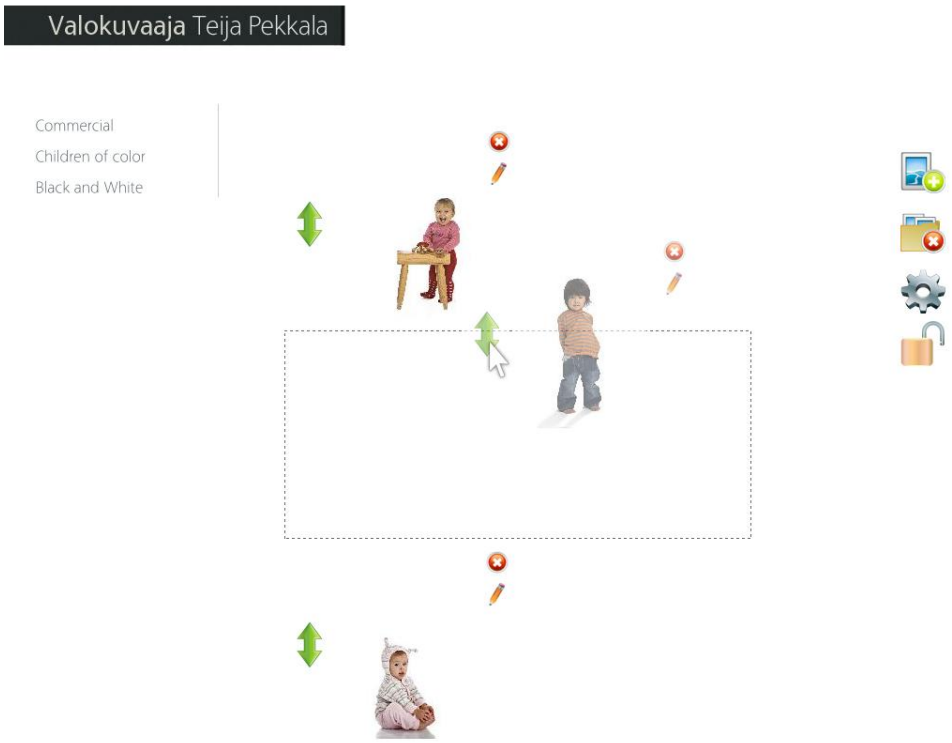
Käyttöliittymän graafinen ilme on hyvin selkeä ja se tukee vahvasti tarkoitustaan. Graafisia elementtejä on hyvin niukasti ja typografia on toteutettu täysin vakiintuneilla selainfonteilla. Merkityksellisimmät graafiset elementit ylläpitäjänäkymässä ovat ikonit, jotka ennen kaikkea selkeyttävät käyttöliittymän toimintaa, mutta myös tuovat siihen omaa visuaalista ilmettä (kuva 24.). Ylläpitäjä näkymä mukailee varsinaista kuvagallerian ilmettä, mutta koska tämän näkymän kohdeyleisö on hyvin suppea voidaan myös visuaalisen suunnittelun tavoitteet määrittää tämän perusteella.

kuva 24. Kuvagallerian ikoneita

Käyttöliittymänäkymän on tarkoitus olla tyyliltään hyvin yhtäläinen varsinaisen yleisen selainnäkymän kanssa. Tämän ansiosta käyttökynnys ylläpitotehtävien opetteluun on pienempi.



kuva 25. Ylläpitäjänäkymän yleisilme on hyvin yksinkertainen ja kaikki toiminnot ovat suoraa käyttäjän tavoitettavissa. Toimintoja selkeyttää myös ikoneiden viereen ilmestyvät opastekstit



kuva 26. Toimintoja on pyritty yksinkertaistamaan käyttämällä mm. drag and drop –ominaisuutta kuvien järjestelemisessä. Käyttäjän siirtäessä kuvaa hänelle ilmestyy vaihtoehtoja uusista kuvan paikoista. Raahauksen lopuksi kuva siirtyy uudelle paikalla ja tämä paikkatieto päivittyy MySQL-tietokantaan.

5.9 Lopputuloksen tarkastelu

Garretin viisitasoinen malli antoi opinnäytetyön toiselle CASE-projektille äärimmäisen toimivan mallin web-pohjaisen sovelluksen toteuttamiseen. Projekti eteni koko työprosessin ajan määrätietoisesti ja lopputulos vastasi alussa asetettuja tavoitteita. Toteutusmuoto oli myös riittävän modulaarinen jatko kehitystä varten. Valmiin mallin ansiosta myös projekti aikataulujen noudattaminen oli helpompaa.

Toimivan kokonaisuuden valmistuttua vastaavissa projekteissa seuraavat laajamittaiset työvaiheet ovat testaus ja käyttöönotto. Tässä opinnäytetyössä keskityttiin ainoastaan www-sisällönhallinnan käyttöön ja sen suunnitteluun, joten näitä osa-alueita ei käsitellä enää opinnäytetyön yhteydessä. On kuitenkin tärkeää ymmärtää, että täysimittainen web-projekti ei lopu varsinaisen tuotteen valmistumiseen.

6. YHTEENVETO

Opinnäytetyön päätarkoitus oli tutkia www-sisällönhallintajärjestelmiä ja kartoittaa tämän hetkistä markkinatarjontaa. Tämän tutkimuksen pohjalta pystytään aloittamaan sisällönhallintajärjestelmän sisältävän sivun toteuttaminen. Suurena haasteena opinnäytetyössä oli hyvin nuoren ja epäkypsän aihealueen käsittely. Asiantuntijoiden näkemys ja tulkintaerot vaikeuttivat oikein ja käytännöllisen tiedon keräämistä tätä opinnäytetyötä ajatellen.

Kirjallisuuslähteiden lisäksi varmuudella ajanmukaista sisältöä haettiin useiden internet-lähteiden kautta joista web-statistiikkaa tarjoavat sivustot osoittautuivat erityisen hyödyllisiksi. Näiden kautta opinnäytetyöhön saatiin kerättyä informaatiota markkinoilla olevista järjestelmistä puolueettomien lähteiden kautta, joidenka tarjoama tieto on varmasti ajanmukaista.

Peruskäsitteiden ymmärtäminen ja asiantuntijoiden poikkeavien toimintamallien tarkka tutkiminen osoittautui välttämättömäksi opinnäytetyön kannalta ja tämä osuus saikin hyvin mittavan osan koko teoriaosuudesta. Projektista riippumatta suunnittelun merkitystä ei voida koskaan vähätellä, joten edellä mainitun osa-alueen merkitys on hyvin kiistaton.

CASE-projektien toteutuksessa käytiin läpi kaksi erilaista sisällönhallintajärjestelmän sisältävää web-sivutoteutusta. Projektien erilaisuus toi opinnäytetyöhön hyvää sisältöä, jossa teoriaosuuden informaatiota pystyttiin hyödyntämään erinomaisesti. Molemmat projekteista onnistui myös erittäin hyvin. Opinnäytetyössä näiden projektien osuus jäi hyvin teoriapainotteiseksi, koska tämä sisältö tuki paremmin koko opinnäytetyön tyyliä ja tarkoitusta.

Kokonaisuutena opinnäytetyö oli onnistunut. Sen avulla onnistutaan selvittämään www-sisällönhallintaan liittyvät aiheet ja toiminnot perusteellisesti liittäen mukaan myös hieman teknisen toteutuksen esimerkkejä. Opinnäytetyö antaa kattavan

pohjan laajempaa tutkimusta varten ja siitä saatua tietoa voi myös hyödyntää muiden kuin www-sisällönhallintajärjestelmien suunnitteluun. Jatkotutkimuksena tämän opinnäytetyön täydennykseksi aiheelta sopisivat laajamittaisten järjestelmien suunnittelu tai perusteellinen järjestelmän käyttöliittymäsuunnittelu.

LÄHTEET

Alexa The Web Information Company 2010 [verkkojulkaisu] [viitattu 10.2.2010] saatavissa http://www.alexa.com

Boiko B. 2005. Content Management Bible, Indiapolis: Wiley Publishing Inc.

eZ Publish the content management ecosystem 2010 [verkkojulkaisu] [viitattu 5.2.2010] saatavissa: http://ez.no/

Friedlein A. 2003. Maintaininh & Evolving Succesful Commercial Web Sites, Morgan Kaufmann Publishers

Garret J. 2002. The Elements of User Experience. American Institue of Graphic Arts

Goodwin S, Vidgen R. 2002. Content content everywhere… time to stop and think? The process of web content management, Computing & Control, Engineering Journal 13 (2). Institute of Electrical and Electronics Engineers

Graf H, 2007, Building Websites with Joomla! 1.5 Beta 1, Iso-Britania Birmingham, Packt Publishing

Joomla Documentation 2010 [verkkojulkaisu] [viitattu 2.3.2010] saatavissa: http://docs.joomla.org/

Joomla translation team 2010 [verkkojulkaisu] [viitattu 27.1.2009] saatavissa: http://www.joomlaportal.fi

Karlsson T, Boije J. 2005. Content management systems – business effects of an implementation, master thesis in Informatics specializing Business Techonology.

Powel W, Gill C, 2003, Web Content Management Systems in Higher Education, (2) Eduacause Quartely

Shreves R, 2008, Open Source CMS Market Share, Water & Stone [viitattu 3.2.2010] . Saatavissa http://www.j-b-t.com.au/pdfs/2008OpenSourceCMSMarketSurvey.pdf.

Tolvanen P. 2008. [verkkolähde] Julkaisujärjestelmät Suomessa, markkinakatsaus 2008 [viitattu 31.3.2008]. saatavissa: http://vierityspalkki.fi/2008/03/31/julkaisujarjestelmat-suomessa-markkinakatsaus-2008/

TYPO3 Core Development Team. 2008. [verkkolähde] TypoScript Syntax and In-Depth Study, 3) [viitattu 5.3.2010]. saatavissa: http://typo3.org/documentation/document-library/core-documentation/doc_core_ts/current/

LIITTEET