Suoravaikutteisuus sisällönhallinnan käyttöliittymissä

Olli Savolainen

Tiivistelmä

Tämä tutkielma käsittelee sisällönhallintajärjestelmien sisällön organisointiin ja muokkaamiseen liittyviä käyttöliittymiä ja käsitteistöä, keskittyen pienehköjen yksityishenkilöiden, yhdistysten ja pienyritysten sivustojen hallintaan. Vertailen sisällönhallintajärjestelmiä Joomla! ja Google Page Creator, ja kokoan osittain niiden pohjalta käsitteistön WWW-sivuston rakenteen ja sisällön hallintatoiminnoille. Esitän suoravaikutteisen käyttöliittymän WWW-sivuston hallintaan.

Avainsanat ja -sanonnat: käyttöliittymä, käsitteistö, sisällönhallinta, WWW
CR-luokat: H.5.2, H.5.4, I.7.1

1. Johdanto

WWW:n käyttäminen ihmisen luovuuden ja kommunikoinnin välineenä on usein kaukana luonnollisesta. Varsinkin WWW-sivujen tekemisestä kuulee puhuttavan ohjelmointina, joka viestii sitä, että ihmiset kokevat kysymyksessä olevan ammattitaitoa vaativan työn. Toisaalta käytettävyyden ja saavutettavuuden näkökulmat jäävät helposti unohduksiin vaikkapa työntekijälle, jolle WWW-sivujen ylläpito on usein annettu palkattomaksi tehtäväksi ”muun ohella”. Tämän tutkielman tarkoituksena on selvittää suoravaikutteisten käyttöliittymien lupausta tehdä WWW-sivustojen hallinnasta helpommin opittavaa.

Ennen sisällönhallinnan käyttöliittymän suunnittelemista on tunnettava informaation, jota halutaan hallita, rakenne. Usein sisällönhallintajärjestelmät (myöhemmin *järjestelmät*) antavat käyttäjän hallittavaksi ikioman käsitteistön ja käyttöliittymänsä. Ihmisellä, jolla on kokemusta WWW:n selailusta, kuitenkin on jo kieli – käsitteistö – sitä koskien, mistä WWW-sivuissa on kyse. Käsitteet kuten sivu, sivusto, osoite, artikkeli/juttu, kuva, valikko/menu ja linkki ovat tuttuja monille. Sisällönhallintajärjestelmiin liittyen olisi kuitenkin tärkeää tutkia laajemmin, mitkä käsitteet todella tunnetaan eritasoisten WWW-sivujen tekijöiden keskuudessa. Toki sisällönhallintaan liittyy myös abstrakteja käsitteitä, kuten sivupohja (engl. template), jotka eivät ole ilmeisiä tavalliselle internetin samoilijalle. Kyse ei siis ole pelkästään sanoista, vaan riittävän ilmaisuvoimaisen käsitteistön löytämisestä ja kehittämisestä siten, että järjestelmä on mahdollisimman joustava käyttäjän tarpeisiin nähden.

Järjestelmien pitäisi mielestäni keskittyä enemmän työn kohteeseen, sivustoon, kuin omaan toiminnallisuuteensa ja käsitteistöönsä. Tämä oli lähtökohtanani lähtiessäni selvittämään sisällönhallintajärjestelmien käyttökynnyksen alentamismahdollisuuksia:

Työpöytäsovelluksissa *raahaaminen* (engl. drag and drop) sekä suoravaikutteiset (engl. direct manipulation) käyttöliittymät ovat olleet tuttuja suurellekin massalle jo pidemmän aikaa. Schneidermanin [1997, 1] mukaan suoravaikutteisilla käyttöliittymillä on monia positiivisia ominaisuuksia, kuten: Aloittelijat voivat oppia perustoiminnallisuutta nopeasti, ja käyttäjät myös hermostuvat ohjelmille vähemmän, koska pystyvät välittömästi näkemään tekojensa tulokset ja ymmärtävät järjestelmän toimintaa paremmin. Suoravaikutteisuus on erittäin modaalista, vaikka vaikuttaakin modaalittomalta [Jacob, 1986, ss. 287-288]: se tekee moodeista helppoja tekemällä ne ilmeisiksi ja helpoiksi vaihtaa.

Huolimatta siitä, että esittämäni sisällönhallintakäyttöliittymä ei jaa sisältöä, toimintalogiikkaa ja ulkoasua erilleen, on ajatuksena, että järjestelmän sisäisesti nämä kaikki ovat selkeästi eroteltuina. Tarkoitus on myös, että samaan tietoon tarjotaan muitakin näkymiä kuin pelkästään esittämäni käyttöliittymä. Esimerkiksi valikoiden suoravaikutteisen muokkaamisen kautta pystytään luomaan sivustolle yksinkertainen rakenne, mutta käyttäjän halutessa luokitella sisältöä monipuolisemmin eri kategorioihin tarvitaan jo erillinen rakennenäkymä. Joka tapauksessa esittämäni käyttöliittymä lienee suoravaikutteisuutensa vuoksi aloitteleville käyttäjille ensisijainen tapa lähteä sivustoa rakentamaan, ja olettaen että järjestelmä sisäisesti säilyttää sisällön, ulkoasun ja logiikan erottelun, ei kokeneemmankaan käyttäjän pitäisi tarvita kajota ”konepellin alle”.

Käyttöliittymän rakentaminen suoravaikutteiseksi tekee sisällön hallinnan helpommaksi tehokäyttäjillekin; kohdattavana ei ole ensisijaisesti uutta sisällönhallintakäyttöliittymää vaan oma, muokattava sivusto. Toki käyttäjän on myös sallittava tarvittaessa muokata suoraan myös kaikkea HTML:ää ja CSS:ää, jota järjestelmä tuottaa – kuitenkin mahdollisuuksien rajoissa säilyttäen erottelun sisällön, esitystavan ja toimintalogiikan välillä.

2. Rajaukset

Tarpeet, joita sisällönhallintajärjestelmien odotetaan täyttävän, ovat erittäin moninaisia. Tarkastelussa ovat sisällönhallintajärjestelmät yksityishenkilöiden, pienyritysten ja yhdistysten käyttöön – toisin sanoen niiden, joilla usein ei ole varaa palkata omaa WWW-ohjelmoijaa. Keskityn pääasiassa avoimen lähdekoodin suhteellisen pienimuotoisiin sisällönhallintajärjestelmiin ja niiden sisällön hallinnan ominaisuuksiin. Projektin tarpeet on joka tapauksessa kutakin yksittäistä sivustoa toteuttaessa arvioitava erikseen. En käsittele kaikkea toiminnallisuutta, jota sisällönhallintajärjestelmässä voidaan odottaa olevan, kuten käyttäjähallintaa tai työnkulunohjausta (engl. workflow). [Robertson, 2002]

Tässä tutkielmassa tarkoitetaan *käyttäjällä* sivuston muokkaajaa. Erotuksena käyttäjästä, WWW-sivuston *kävijä* ei yleensä pysty muokkaamaan sivustoa, tai saattaa pystyä tekemään niin ainoastaan rajoitetusti. Kävijän ja käyttäjän ero hämärtyy edelleen

Wiki-ohjelmistoa käyttävissä sivustoissa ja muissa verkkosovelluksissa, joita ei kuitenkaan tässä tutkielmassa käsitellä.

Tutkielman vertailuosassa etsin erilaisia lähestymistapoja kysymykseen siitä, minkälaisia käyttöliittymillä ja käsitteistöillä sivuston sisältöä ja rakennetta voi hallita. Olen jättänyt monia järjestelmiä, jotka tarjoavat käsiteltyjen ohjelmistojen kanssa samankaltaisia ratkaisuja, vertailun ulkopuolelle. Vertailemistani järjestelmistä Joomla! on avoimen lähdekoodin ratkaisu ja Google Page Creator ilmainen WWW-palvelu. Omat mahdollisuuteni päästä käsiksi kaupallisiin järjestelmiin ovat tutkielmaa tehdessä olleet rajoittuneet, samoin kuin kohdekäyttäjäryhmillä usein. Aiheeseen pidemmälle syventyessä olisi hyvä saada mukaan suljettujakin järjestelmiä.

Huomion kohteena on pääasiassa HTML-/CSS-pohjaisen sisällön tuottaminen ja hallinta, joskaan mikään esitetty ei poissulje muiden esitysteknologioiden käyttöä. Ainakin sitä osaa sivuston CSS:stä, joka ei määrää sivuston asettelua vaan yksittäisten elementtien muotoilua, on käyttäjän pystyttävä muokkaamaan tuntematta CSS:ää, samoin kuin jonkin yksittäisen sivun sivupohjasta erillään olevia CSS-sääntöjä. CSS:n muokkaaminen suoravaikutteisesti on nähdäkseni vielä ratkaisematon haaste, enkä käsittele sitä tässä tutkielmassa. Myöskään *rakennenäkymäksi* kutsumaani käyttöliittymää, eli sisällön luokitteluun ja metadatan hallintaan tarkoitettua käyttöliittymiä en esitä tässä vaiheessa.

WWW-sivujen varsinaisen ulkoasun ja sivupohjien (engl. template) toteutus on edelleen mielestäni ammattilaisten tehtävä, eikä suunnittelemani käyttöliittymä ole tarkoitettu siihen. Markkinoilla on kyllä WYSIWYG-työkaluja joilla voi tehdä sivupohjan HTML:ää ja CSS:ää tuntematta, mutta näiden käyttämisessä on yleensä vakavia rajoituksia tuloksen laadussa. Toisaalta pienyritys- ja yhdistyskäytössä tämän ei tarvitse välttämättä tarkoittaa sitä, että sivupohjasta on maksettava: esim. tarkastelemani Google Page Creator tarjoaa käyttäjilleen suuren määrän valmiita pohjia.

3. Haasteet

Lähtiessä tarkemmin hahmottamaan pohjaa, jolle suoravaikutteinen sisällönhallintakäyttöliittymä pitäisi suunnitella, päädyin seuraaviin vaatimuksiin:

1. Käyttäjän pitäisi pystyä WWW-sivustoa rakentaessaan käyttämään mielikuvitustaan vapaasti.
2. Järjestelmän pitäisi pystyä ohjeistamaan käyttäjää tilanteissa, joissa hän toteuttaa mielikuvitustaan olemassa olevien WWW-käytäntöjen tai käytettävyyssuositusten ohi. Toisaalta järjestelmän pitäisi antaa useimpiin tilanteisiin järkevät oletusarvot siten, että käyttäjän ei tarvitsisi kiinnittää huomiota yksityiskohtiin, jotka eivät suoraan liity hänen tavoitteisiinsa.

3. Käyttäjän pitäisi pystyä toteuttamaan visionsa pitkälti suoravaikutteista käyttöliittymää käyttäen, joutumatta ensin opettelemaan ennestään tuntematonta sisällönhallintakäsitteistöä.

Ei vielä riitä, että käyttäjä pystyy rajattomasti käyttämään mielikuvitustaan sivustoaan tehdessä. Mikäli sivustolla on kaupallisia tai muita tavoitteita, halutaan yleensä muun muassa, että tieto löytyy helposti, ja että sivustolle on pääsy mahdollisimman monenlaisilla selaimilla. Sivuston käytettävyydestä ja saavutettavuudesta on huolehdittava myös silloin, kun sivuston tekijällä ei ole käytettävyyskoulutusta.

Kysymys kuuluu siis: Kuinka paljon suosituksia voitaisiin pienten ohjeiden muodossa tai muina käyttöliittymäelementteinä sisällyttää sivuston ylläpitokäyttöliittymään itseensä? W3C toteaa työvedoksessaan *Techniques For Accessibility Evaluation And Repair Tools*: ”On välttämätöntä, että millä tahansa työkalulla on ominaisuuksia, jotka tukevat muistuttamalla nalkuttamatta; auttamalla alentumatta; ehdottamalla esittämättä vaatimuksia” [W3C, 2000b, lainaus käännetty]. Sloan et al. [2000, 2], kuten myös Mankoff et al. [2005, 42], toteavat, ettei pelkkien suositusten seuraaminen (esim. WGAC [W3C, 1999] ja AERT [W3C, 2000a]) riitä – tarvitaan heuristista arviointia ja jopa testausta. W3C:n tarjoamien resurssien lisäksi W4A – International CrossDisciplinary Workshop on Web Accessibility (www.w4a.info), ja varsinkin vuoden 2005 työpajaraportti [Harper et al., 2005] tarjoavat hyviä resursseja lähdettäessä rakentamaan sisällönhallintaohjelmistoa, joka tukee käyttäjää käytettävien sivustojen rakentamisessa. Tämän tutkielman puitteissa en kuitenkaan käsittele automaattista käytettävyyden arviointia tarkemmin.

Se, mitä WWW-sivuilta yleensä odotetaan, ilmaistaan usein vapaamuotoisesti, vaikkapa seuraavaan tapaan:

- ”Haluaisin sivustolle sellaisen *uutispalstan*,
- ja tuossa (osoittaa kohtaa sivulla) voisi olla käyttäjän huomiota herättämään se *kuva*, joka saatiin messuilta ja se *juttu*, jonka Jaska kirjoitti,
- ja laitetaan tuohon oikeaan reunaan *kysely*, että messuilla käyneet voivat äänestää ja kommentoida, oliko tämän vuoden messuedustus hyvä.”

Tarkoituksena on, että jo tällaisesta tarpeiden kuvauksesta olisi helppoa siirtyä tekemään sivustoa, raahaten paikalleen haluttuja elementtejä (esimerkissä uutispalsta, kuva, juttu, kysely). Tavoitteena on käyttöliittymä, jonka avulla voidaan paitsi määritellä sivusto, myös tehdä helpoksi koota muiden, erikoistuneiden sisällönhallintaohjelmistojen tai –komponenttien paremmin tuottama sisältö saman sivuston alle. Suoravaikutteisuuden käyttöliittymään sisään rakentaminen näyttäisi suovan tähän mahdollisuuden.

4. Semantiikka ja suoravaikutteinen muokkaus

4.1. Sisällön järjestelemisestä suoravaikutteisesti

Sivuston sisältö kuvataan lopulta HTML:n avulla, joka on semanttinen kieli – eikä siis tarkoitettu ulkoasun määrittelyyn). Esitän tässä tutkielmassa käyttöliittymän sivuston muokkaamiseen suoravaikutteisesti, toisin sanoen olennaisesti visuaalisella tavalla, ja tämä näyttää olevan ristiriidassa HTML:n semanttisuuden kanssa. Herää kysymys: jos jotain muokataan visuaalisesti, eikö pitäisi muokata mieluummin CSS:ää kuin HTML:ää? Vastaus kuuluu: kyllä ja ei.

Ongelma HTML:n muokkaamisessa visuaalisesti on se, että HTML on pohjimmiltaan yksiulotteinen esitystapa, kun taas WWW-sivu esitetään (pääasiassa) kaksiulotteisena. HTML-puussa ensin esiintyvä sisältö voi CSS:n määrittelemässä ulkoasussa olla vierekkäin myöhemmin sisältyvän sisällön kanssa. Pelkkää elementtien horisontaalista sijaintia katsomalla ei ole mahdollista päätellä varmasti elementtien järjestystä HTML-puussa.

Sisällönhallinta on toisaalta enemmän suurempien kokonaisuuksien, hallintaa. Kuvailtu ongelma ilmenee ainoastaan yksittäisten sivujen tasolla, ei sivuston tasolla sisältöä organisoidessa.

4.1.1. Esimerkki

Yksinkertaisen esimerkkisivustomme suunnittelija on oikeaoppisesti suunnitellut sivupohjan HTML:n miettien rakennetta ja semantiikkaa, ja asettanut dokumentin tärkeimmän – varsinaisen sisältöalueen – sivupohjaan alkuun. HTML-dokumentissa elementit ilmenevät siis järjestyksessä: sisältö, valikko, mainokset. Graafikko, joka ideaalitilanteessa on tekemisissä ainoastaan CSS:n ja visuaalisten elementtien kanssa, on suunnitellut sivustolle kolmipalstaisen ulkoasun: Hän laittaa valikon aivan vasemmalle, sisällön keskimmäiseen palstaan ja mainokset oikeaan palstaan. Sivupohja toimitetaan sisällönhallintajärjestelmään, jossa toimittaja voi sitten ylläpitää sivustoa.

Eräänä päivänä toimittaja saa johtajalta määräyksen, että oikein näkyville, sivuston oikeaan palstaan tuleekin nyt kävijöiden mieltymyksiä mittaava kysely. Toimittaja siirtää mainokset oikeasta palstasta vasempaan, valikon alle vähemmän näkyville, ja laittaa kyselymoduulin oikean palstan huipulle. Tulos: HTML:ssä elementtien järjestys on nyt: sisältö, valikko, mainokset, kysely. Mobiililaitteiden tai ääniselainten käyttäjät harvemmin löytävät tärkeäksi tarkoitettua , mutta heille nyt sivun pohjalla näkyvää, kyselyä.

4.1.2. Lähestymistapoja

Ongelmatilanteita varten järjestelmän on vähintään tarjottava mistä tahansa sivusta näkymä ilman CSS:ää – myöntäen tosin, että tämä ratkaisu lähinnä kiertää ongelman. Esimerkkimme toimittaja pystyisi nyt näkemään elementtien todellisen järjestyksen, ja

vaikka siirtämään mainokset vasemman palstan pohjalta oikean palstan pohjalle, jolloin mainittu järjestys olisi: sisältö, valikko, kysely, mainokset.

On huomattava, että kyse on toisaalta myös CSS:n itsensä asettamista rajoituksista: ulkoasun, jossa yhdessä palstassa olevien elementtien välissä olisi HTML-puussa toisen palstan elementtejä, määrittäminen on CSS-kielellä hyvin monimutkaista, ellei mahdotonta.

Ongelma ei kuitenkaan tule aina esille, vaan usein visuaalinen esitysjärjestys vastaa yksi yhteen HTML-puun järjestystä: Suurin osa WWW-sivun sisällöstä on miellettävissä jonkinlaisiksi listoiksi. Tämä pätee ainakin valikoihin ja sisältökatkelmien listauksiin: siirrettäessä listan elementtiä WYSIWYG-käyttöliittymässä, voidaan sitä siirtää vastaavasti myös HTML:ssä.

4.2. Tyylien suoramuokkaus

WYSIWYG-käyttöliittymissä (Mitä näet on, mitä saat) ongelmaksi muodostuu usein myös ”What You See Is All You Get” – mitä näet on **kaikki**, mitä saat. Tekstisisällön muotoiluihin liitettynä tämä ongelma ilmenee siinä, että vaikka yksittäistä tekstiä onkin helppo muotoilla, niin varsinaisten CSS-tyylien käsittely – vaikkapa kaikkien dokumentin tai sivuston otsikkojen muotoilu kerralla – on suoravaikutteisesti vaikeaa. Suuri osa tätä ongelmaa on se, että ennen kuin käyttäjä voi oppia käyttämään tyylejä, hänen täytyy tuntea niiden toiminta-ajatus. Yksi suoravaikutteisuutta lähestyvä tapa määritellä tyylejä on *esimerkin kautta* [Myers, 1991]. Yksinkertaistettuna idea on seuraava: Ensin muotoillaan jokin teksti halutunlaiseksi, ja sitten painetaan ”luo uusi tyyli” –painiketta, jolloin kyseinen muotoilu – tyyli – on käytettävissä minkä tahansa muun tekstin muotoiluun. Tämä ominaisuus on sekä Microsoft Word 2003:ssa että OpenOffice 2 Writer:ssä, ja toivoisin, että tulevaisuudessa tämä ominaisuus saataisiin myös selainpohjaisiin WYSIWYG-muokkaimiin.

5. Vertailu

Seuraavassa tarkastelen kahden sisällönhallintajärjestelmän, Joomla!:n ja Google Page Creatorin käyttöliittymiä ja käsitteistöä, saavuttaen näkemyksen siitä, minkälaisia käyttöliittymäratkaisuja sivuston sisällön ja rakenteen hallinnassa tänä päivänä on käytössä.

Vertailluissa järjestelmissä on kiinnitetty huomiota käytettyyn käsitteistöön ja siihen, kuinka vapaasti ja kuinka suoravaikutteisesti käyttäjä voi muokata sivuston rakennetta ja sisältöä.

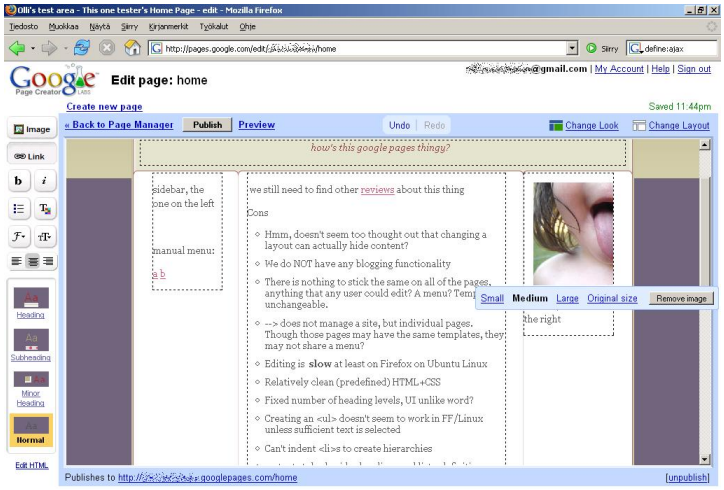
5.1. Google Page Creator

5.1.1. Käyttöliittymä

Googlen keväällä 2006 julkaisema työkalu on merkittävä askel suoravaikutteisuuden tuomisessa WWW-sisällönhallintaan. Vahvasti AJAX-teknologiaan (engl. Asynchronous JavaScript and XML) nojaava Google Page Creator (myöhemmin GPC) tuo etualalle sivun, jonka sisältöä käyttäjä voi WYSIWYG-tyylisesti muokata.

Erona useimpiin muihin järjestelmiin on se, ettei käyttäjälle anneta WYSIWYG-näkymää ainoastaan rajoitettuun tekstikentässä näkyvään sisältöön, vaan koko sivun muokkaukseen. GPC:n lähestymistavassa sivu on jaettu kenttiin, joista jokaisen sisältöä käyttäjä voi muokata. Suurin etu tästä on se, että sisältöä (kuvia, tekstiä) voi siirrellä sivun sisällä kentästä toiseen raahaamalla. Klikkaamalla kuvaa tai linkkiä aukeaa niille pieni ponnahdusvalikko, josta kuvaa käsiteltäessä voi muuttaa sen kokoa tai poistaa sen ja linkkiä käsiteltäessä voi vaihtaa linkin kohdetta. Myös kuvien asemointi vasemmalle, oikealle tai keskelle onnistuu vetäessä niitä hiirellä. Suoravaikutteisuuden puolesta tärkeitä, WWW-pohjaisissa sovelluksissa aiemmin harvinaisia ominaisuuksia ovat myös automaattinen tallennus sekä tehtyjen toimintojen kumoaminen ja toistaminen (engl. undo, redo).

Yksittäisen sivun yleisulkoasua käyttäjä voi muokata valitsemalla valikoimasta *lookeja* (vastaa käyttämäni käsitteistön sivupohjaa) sekä *layouteja* (kenttien sijainnit *lookissa*).



Kuva 1. Google Page Creatorin suoravaikutteinen muokkauskäyttöliittymä ja oikealla kuvan ponnahdusvalikko kuvan ominaisuuksien muokkaamiseen

5.1.2. Sivuston rakenteen hallinta

Kuten nimikin kertoo, GPC ei kuitenkaan ole tarkoitettu sivuston, vaan pääasiassa yksittäisten sivujen hallintaan – tässä mielessä se muistuttaa enemmän Wikiä kuin sisällönhallintajärjestelmää. Lookeja ja layouteja lukuun ottamatta ei eri sivuilla ole

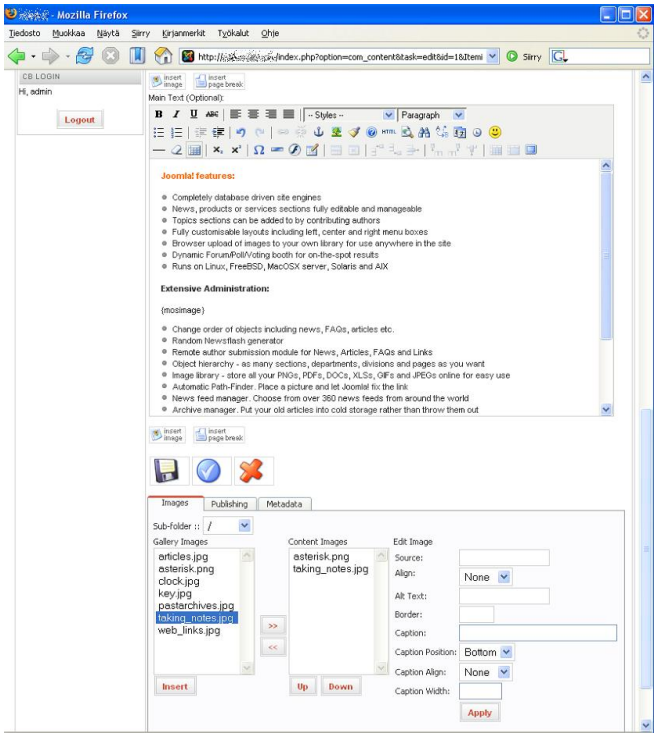
mitään yhteisiä elementtejä. Edes valikkoa, joka mahdollistaisi sivujen välisen navigoinnin, ei ole mahdollista rakentaa.

5.2. Joomla! 1.08

Joomla!, entinen Mambo Open Source, on yksi suosituimmista open source -sisällönhallintajärjestelmistä. Olen valinnut sen tarkasteluun sen monipuolisuuden ansiosta nimenomaan kokonaisen sivuston hallinnasa.

5.2.1. Käyttöliittymä

Joomla!:n hallintakäyttöliittymä ei ole suoravaikutteinen, vaan eri sivuston sisällön ja rakenteen hallinnan työkalut on jaettu hallintakäyttöliittymässä ikkunointijärjestelmistä tutuin alasvetovalikoin. Hallintapaneelin lisäksi tarjotaan kuitenkin mahdollisuus kirjautua sisään myös sivustolla itsellään ja muokata jo valmiiksi sivuilla olevia sisältöjä. Kaikki muut varsinaisen sivuston ylläpitoon liittyvät toiminnot on hoidettava erillisen hallintapaneelin kautta, joskin erillisiä komponentteja kuten keskustelupalstaa voi käyttää sisäänkirjautuneena.



Kuva 2. Sisällönmuokkauskäyttöliittymä Joomla!:ssa: teksti on muokattavissa WYSIWYG-tyyllisesti, mutta kuvien lisäämiseen käytetään erityistä syntaksia ja kuvat valitaan erillisen, valikkokäyttöliittymän avulla (kuvan alaosa).

5.2.2. Sisällön rakenteen hallinta

Joomla! sisältää erilliset komponentit uutissisällön ja staattisen sisällön muokkaamiseen. Uutissisältö organisoidaan sektioihin ja kategorioihin [Open Source Matters, 2006]:

kaiken uutissisällön on oltava tarkalleen yhdessä kategoriassa joka on tarkalleen yhdessä sektiossa. Tämän lisäksi voidaan käyttää staattista sisältöä, joka on uutissisällöstä erillään eikä sitä voi järjestää hierarkiaksi. Joomla!:n valikkomuokkain määrää sivuston käyttäjälle näkyvän hierarkian: valikon mistä tahansa kohdasta voi tehdä linkin uutissisältöön, staattiseen sisältöön, dynaamisen komponentin generoimaan sisältöön tai sivuston ulkopuolelle.

Uutissisällön kategorisoinnin hyötynä on se, että Joomla! osaa tuottaa erilaisia listauksia, esimerkiksi verkkopäiväkirjan tyyppisiä, niiden perusteella. Staattisen sisällön käyttö on yksinkertaisempaa, mutta samalla sen rajoitteena on se, että sitä ei voi luokitella, eikä Joomla! osaa automaattisesti listata staattista sisältöä.

Joomla!:n sivupohja käsittävät myös sivustolle yhteisen CSS:n. Sivupohjaan asetetaan eri moduuleja. Joomla!:ssa sivupohjan positioihin asetettavat moduulit ovat jokaisella kyseistä sivupohjaa käyttävällä sivulla samat, vaikkakin moduuleille voi määrätä erikseen, millä sivuilla ne näkyvät.

Joomla!:ssa on dynaamisesti toimiville, periaatteessa itsenäisille ja myös kolmannen osapuolen lisäosina asennettaville yksiköille kolme eri käsitettä: *komponentti*, *mambot* ja *moduuli*. Toisinaan tietyn toiminnallisuuden tarjoava paketti voi vaatia moniakin näistä:

Komponentteja (engl. component) ovat yleensä lisäosat, jotka tuottavat sisältöä sisältöalueelle, kuten vieraskirja, kuvagalleria tai Joomla!:ssa jopa etusivu.

Moduuli (engl. module) on sivun osaan sijoitettava ”laatikko”, jota käytetään sisällön esittämiseen. Moduuleja ovat muun muassa valikot, breadcrumb, kyselyt (engl. *poll*) tai rekisteröityneiden käyttäjien sisäänkirjautumislomake.

Mambot on pieni ohjelma, joka ajetaan välittömästi ennen minkään sisällön näyttämistä WWW-sivulla. Kyseinen ohjelma voi muokata lopullista sivua, ja usein se korvaa erityisiä merkkijonoja tai “tageja” jonkinlaisella toiminnallisuudella tai muotoilulla. Esimerkiksi kuvien näyttämiseen sisällössä käytetään useimmiten {mosimage}-syntaksia, jonka käsittelee tätä varten tehty mambot.

Erottelu moduulien ja komponenttien välillä on nähdäkseni lähinnä toteutustekninen, eikä vaikkapa vieraskirjalla (komponentti) ja valikolla (moduuli) ole sen kannalta, miten käyttäjä niitä sivuston hallintakäyttöliittymässä käsittelee, perustavanlaatuista eroa. Olenkin käsitteistöehdotuksessani yhdistänyt moduulin ja komponentin käsitteet yhdeksi. Toteutuksessa on toki otettava huomioon, miten moduulin käyttäytyminen ohjelmakoodin tasolla vaihtelee moduulin position vaihtuessa.

Mambotin käsite voidaan nähdäkseni useimmissa tapauksissa tehdä tarpeettomaksi suunnittelemalla käyttöliittymä suoravaikutteiseksi. Edellä mainitun esimerkin {mosimage}-syntaksia ei tarvita, mikäli kuvia voi yksinkertaisesti raahata sisältöön ja ne näkyvät välittömästi.

5.3. Vertailun johtopäätös

Google Page Creator on tämänhetkisistä WWW-pohjaisista järjestelmistä selvästi suoramuokkausta eniten hyväksi käyttävä. Kuitenkaan siihen ei kuulu sivuston rakenteen hallintaan liittyviä ominaisuuksia. Suoravaikutteisuuden lisäksi ehdotuksellani on yhteistä GPG:n kanssa käsite layout, joka mahdollistaa yksittäisten sivujen sisältöalueiden jaottelun siten, että eri alueita voidaan käsitellä erikseen.

Joomla! sisältää laajan valikoiman sivuston rakenteen ja sisällön hallinnan työkaluja. Sen ympärille on kuitenkin luotu käsitteistö, joka osittain nousee järjestelmän teknisestä rakenteesta käyttäjän tarpeiden sijaan, ja tämä nostaa oppimiskynnystä. Joomla!:n käyttöliittymä on valikkopohjainen eikä käytä suoravaikutteisuutta hyväkseen. Ehdotuksessani oleva moduulin käsite on pitkälti yhdistelmä Joomla!:n komponentin ja moduulin käsitteistä. Käsitteistö- ja käyttöliittymäehdotuksessani kehitän Joomla!:n lähestymistapaa sivupohjiin, moduuleihin ja niiden väliseen suhteeseen *Designin* käsitteen muodossa.

6. Ehdotus sisällönhallinnan käyttöliittymäksi ja käsitteistöksi

6.1. Taustaa

Suoravaikutteisuudesta huolimatta eri elementeillä on oltava nimet, joiden kautta käyttäjä voi hahmottaa niiden toimintaa. Ehdotan tässä käsitteistöä, joka tarvitaan WWW-sivuston rakennusaineiksi. Pidän kuitenkin itse käsitteitä sanojen takana tärkeämpinä kuin niiden nykyisiä nimiä. Eri sanojen soveltuvuutta ja ymmärrettävyyttä laajemmalle yleisölle pitäisi tulevassa tutkimuksessa selvittää tarkemmin. Esimerkiksi käyttämääni termiä ”sisältökatkelma” vastaa arkikielessä usein sana ”juttu” tai ”artikkeli”.

Vaikka käyttöliittymä käyttää melko laajaa käsitteistöä, ovat useimmat käsitteet jo muissa järjestelmissä ja WWW-suunnittelijoiden keskuudessa käytössä. Osan käsitteistä, esim. moduuli, merkitys tarkennetaan käsittämään juuri esittelemääni järjestelmään sopivaksi.

6.2. Käyttöliittymän yleiskuvaus

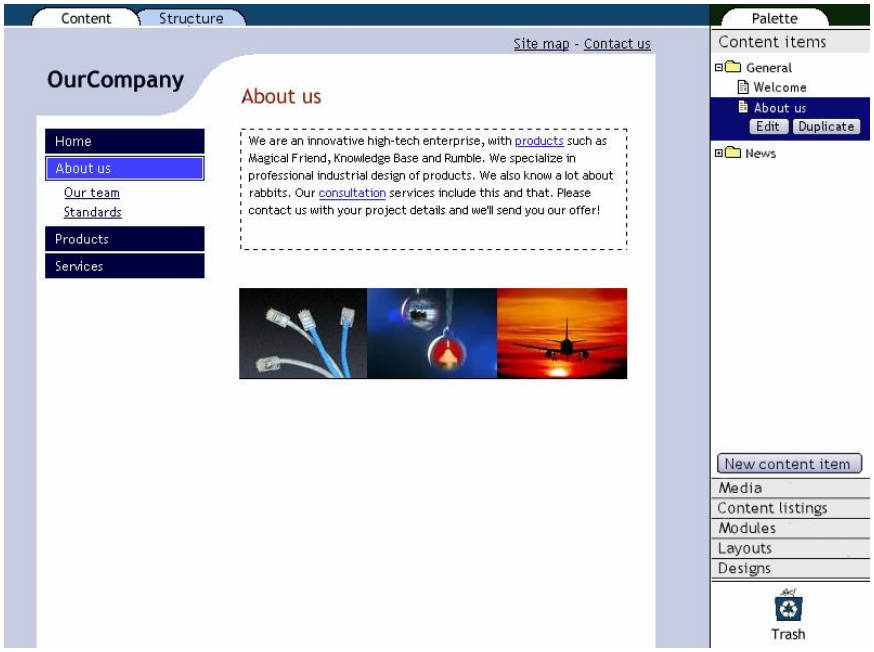
Käyttöliittymä jakautuu muokkausalueeseen, roskakoriin sekä palettiin, jonka avulla sisältöä ja sivuston rakennetta hallitaan, ja josta elementtejä voi raahata muokkausalueelle. Muokkausalueen ensisijainen näkymä, jonka esittelen tämän tutkielman puitteissa, on sivusto kävijälle näkyvässä muodossaan. Perusajatuksena tässä näkymässä on se, että käyttäjä voi raahata minkä tahansa sisältöelementin joko valikkoon (luoden uuden kohdan menuun ja uuden sivun, jolla on raahattu elementti sisältönä), sisältöalueelle tai roskakoriin. Lisäksi käyttäjä voi muokata näkyviä sisältöelementtejä WYSIWYG-tyyliin kuten vertailussa esitellyssä Google Page Creator:ssa. Järjestelmä

tukee käyttäjää semanttisen työskentelytavan omaksumisessa: tekstityylit, joita käytetään, olisivat ihannetapauksessa kaikki määriteltyinä CSS-tiedostossa, ja käyttäjä yksinkertaisesti käyttäisi WYSIWYG-editorissa näitä määriteltyjä tyylejä.

Koska paletin eri osiot on jaoteltu määrittelemieni sisällönhallinnan peruskäsitteiden mukaisesti, ei käyttäjän tarvitse opiskella koko käsitteistöä heti pystyäkseen hoitamaan perustehtäviä: Pelkän tekstisisällön hallintaan riittää paletin sisältökatkelmia koskevassa osiossa pysytteleminen. Käyttäjän halutessa lisätä sisällön sekaan kuvia, voi hän siirtyä Media-osioon, joka käyttöliittymältään on lähes identtinen sisältökatkelmaosion kanssa. Käytön opettelu on sikälikin helppoa, että vaikka käsitteistö on osin abstraktia, on tärkeimmät käsitteet järjestetty ikään kuin sanakirjaksi: paletin osioksi. Niitä opetellessa käyttäjän on helppo nähdä, kuinka suuren osan järjestelmän toiminnallisuudesta hän jo hallitsee. Käyttöliittymä on suoravaikutteinen: sivusto rakennetaan raahaamalla elementtejä yleensä paletista sisältöalueelle ja menuun. Elementtejä voi myös siirrellä positiosta toiseen samaan tapaan.

Tulevaisuudessa, kun käyttöliittymä on osana järjestelmää, jossa on ACL (engl. Access Control Lists) eli oikeuksienhallinta, voidaan eritasoisille käyttäjille rajata käyttöoikeudet eri paletin osioihin.

6.3. Käyttöliittymän pääelementit



Kuva 3. Ehdotetun käyttöliittymän perusnäkymä. Valittuna paletissa ’About us’ -tekstikatkelma

6.3.1. Paletti (engl. Palette)

Alue käyttöliittymässä, josta löytyvät kaikki sivuston rakentamiseen käytettävissä olevat elementit ja josta käsin elementtejä voi myös hallita. Paletin päätarkoitus on se, että käyttäjä voi raahata siitä eri elementtejä sivupohjaan tai sivulle. Paletin osiot (eli elementit):

- Sisältökatkelmat (engl. Content items)
- Media
- Sisältölistaukset (engl. Content listings)
- Moduulit (engl. Modules)
- Designit
- Asettelut (engl. Layouts)

Kaikissa näissä paletin osioissa on mahdollista valita listasta tietty elementti, jolloin sen alle tulee näkyviin painike, joka avaa kyseisen elementin ominaisuuksien muokkauksen mahdollistavan modaalisen dialogin. Valitun elementin alla voidaan näyttää lisätietoja kyseisestä sisältöelementistä. Tämän lisäksi vieressä on toinen painike, jonka avulla kyseisestä elementistä voi tehdä kopion, jota voi sitten muokata erikseen.

Paletin jokaisen osion alareunassa on myös uuden elementin lisäämisen kyseiseen osioon mahdollistava painike. Tämän lisäksi paletin Designs-osiossa on sivupohjien hallintapaneelin avaava *hallitse sivupohjia* (engl. manage templates) –painike, joka avaa modaalisen dialogin kyseiseen tarkoitukseen.

6.3.2. Roskakori (engl. Trash)

Roskakoriin voi raahata elementtejä mistä tahansa. Paletista roskakoriin raahattuna mikä tahansa elementti poistuu paletista, ja siis koko järjestelmästä, jääden ainoastaan roskakoriin. Muokkausalueelta roskakoriin raahattuna elementti ainoastaan häviää kyseisestä kohtaa sivua, oli kyse minkälaisesta elementistä tahansa, mutta jää vielä palettiin. Valikon kohtaa muokkausalueella olevasta valikosta raahattuna poistuu sivustolta koko kyseinen sivu, eli asettelun ja siihen asetettujen sisältöelementtien yhdistelmä – näitä vastaavat elementit jäävät kuitenkin edelleen palettiin.

Roskakoria klikkaamalla tulee näkyviin modaalisena dialogina roskakorin sisältö, josta minkä tahansa sisällön voi palauttaa. **Avoin kysymys:** epäselvänä tapauksena tilanne, jossa paikassa, jossa roskakorista palautettava sisältö oli alun perin, on jo jotain muuta.

6.4. Organisointielementit ja –käsitteet

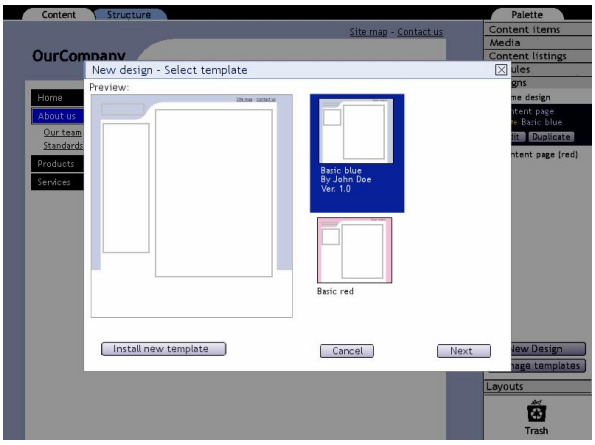
Sivustolla ei ole muuta sisältöä, kuin sisältöelementit, jotka paletti tarjoaa. Kun käyttäjä raahaa saman sisältöelementin kahdelle eri sivulle tai ehkä jopa kahteen eri positioon samalla sivulla, on kyseessä silti vain yksi sisältö, ja yhtä sen näkyvää ilmentymää sivustolla muokattaessa muuttuu myös toinen. Sisältöelementtien lisäksi elementtejä ovat

organisaatioelementeistä asettelut ja designit, jotka ottavat kantaa sivujen rakenteeseen (HTML) ja ulkoasuun (CSS).

6.4.1. Positio (engl. Position)

Jokainen sivuston sivu jakaantuu positioihin. Positiot voidaan jakaa useampiin positioihin asettelujen avulla. Sisältöalue on erityinen positio: siinä näkyy kunkin sivun pääsisältö, jonka mukaan muun muassa kyseisen sivun <title> -elementti ja metadata määräytyy.

Teknisesti positio on paikka HTML-dokumentin XML-puussa, johon sisältöelementtejä voidaan raahata. Positiot määrätään erityisen sivupohjakielen avulla sivupohjassa tai asetteluissa.



Kuva 4. Esimerkki modaalisesta dialogista ehdotetussa käyttöliittymässä: käynnissä uuden Designin rakentaminen, sivupohjan valinta uuteen designiin

6.4.2. Asettelu (osio paletissa; engl. Layout)

Asetteluja voi käyttää sivun sisältöalueen tai muun sivupohjassa määritellyn position jakamiseen osiin tai palstoihin, joihin voidaan raahata eri sisältöjä. Se on siis työkalu, jolla käyttäjä voi määrätä yksittäisten sivujen asettelua vielä designiä hienojakoisemmin ja toisaalta haluttaessa eri sivujen kesken yhtenäisesti. Eri asettelut voivat olla joko sivupohjakohtaisia tai käytössä sivupohjasta riippumatta. Oletuksena positioita ei ole jaoteltu asetteluin, vaan kaikki positioon raahatut elementit menevät positioihin peräkkäin ja allekkain.

Sisäisesti asettelut määrittelevät HTML-katkelman ja CSS-koodin, ja kyseisen HTML-katkelman XML-puuhun positioita sivupohjakielellä. Asettelun voi siis ajatella pienimuotoisena sivupohjana.

Mikäli jokin asettelu raahataan jo olemassa olevan sivun päälle, muuttuu kyseisen sivun sisältöalue tämän asettelun mukaiseksi, ja sivulla jo olleet sisällöt sijoitetaan uuden asettelun positioihin. Mikäli sivulle, jolla on monipalstainen asettelu ja kaikissa palstoissa sisältöä, raahataan asettelu, jossa on vähemmän palstoja kuin alkuperäisessä, on osa palstojen sisällöstä sijoitettava peräkkäin (allekkain) uuden asettelun palstoihin.

6.4.3. Design (osio paletissa)

Design on se osa jokaista sivua, joka on koko sivustolle yhteistä. Jokaisella designillä on yksi sivupohja. Sivupohjan lisäksi design määrittää, mitkä moduulit ovat sivupohjan positioissa: sama sivupohja voi sisältyä moneen designiin, joissa moduulit tai niiden järjestys sivupohjan positioissa vaihtelevat.

Design on mahdollista raahata paletista sivun muokkausnäkymässä sivun päälle, jolloin sivu muuttuu kyseisen designin mukaiseksi. Tarkoituksena on kuitenkin, että designejä asetettaisiin suuremmalle joukolle sivuja kerralla vielä puuttuvassa sivuston rakennenäkymässä.

6.4.4. Sivupohja (engl. Template)

Sivupohja käsittää CSS-tiedoston, sekä HTML-pohjan, johon on jonkin sivupohjien tekoon suunnitellun kielen (engl. template language) avulla määritelty positiot (engl. positions), joihin sisällönhallintajärjestelmä voi syöttää sisältöä. Se osa CSS:stä, joka ei käsittele niinkään eri elementtien asemointia kuin niiden ulkoasua, voidaan tehdä käyttäjälle muokattavaksi helppokäyttöisen lomakekäyttöliittymän kautta.

Sivupohja ei ole oma elementtinsä käyttöliittymässä, vaan se on aina jonkin *designin* osana. Kuten mainittua, moni design saattaa käyttää samaa sivupohjaa.

6.5. Sisältöelementit ja -käsitteet

Sisältöelementit ovat elementtejä, joita voi pudottaa sisältöalueen tai sivupohjan positioihin: sisältökatkelma, moduuli, mediasisältö ja sisältölistaus.

6.5.1. Sisältölistaus (osio paletissa; engl. Content listing)

Sisältölistaus on erityinen dynaaminen elementti, jota nimensä mukaisesti käytetään sisältöelementtien listaukseen. Listaus on dynaaminen: listauksessa ovat sisältöelementit, joille käyttäjä on antanut tietyn luokituksen, järjesteltynä käyttäjän valitseman kriteerin, esimerkiksi päivämäärän, mukaan. Uutissivun tai verkkopäiväkirjan (engl. blog) voi tuottaa sisältölistausta käyttäen. Sisältölistaus voi listata mitä tahansa (muita) sisältöelementtejä.

Sisältölistaukset ovat pohjimmiltaan suotimia: sisältöelementin tai sen metatiedon perusteella voidaan määrittää hakukriteerit ja kentät, joiden perusteella sisältölistaukseen tulevat sisältöelementit järjestetään. Käyttäjälle olisi hyvä tarjota eri mekanismeja sisältölistauksen määrittelemiseen riippuen siitä, kuinka osaava käyttäjä on kyseessä: Yksinkertaisimmillaan sisältölistausta luodessa voidaan kysyä käyttäjältä, mistä sisältökategoriasta sisältö valitaan ja minkä määreen (päivämäärä, aakkosjärjestys jne.) mukaan se järjestetään. Osaavan käyttäjän pitäisi kuitenkin pystyä käyttämään mielikuvitustaan, äärimmäistapauksessa käyttäjän pitäisi siis pystyä käyttämään jonkin ohjelmointikielen suomaa ilmaisuvoimaa. Wolber, Su ja Chiang ehdottavat WYSIWYG-

lähestymistapaa tietokantahakujen tekoon, jota voisi ehkä soveltaa sisältölistausten luontiin [Wolber et al., 2002].

Avoin kysymys: Sekä se, miltä yksittäiset sisältöelementit näyttävät sisältölistauksessa, että se, miltä sisältölistauksesta linkitetyt näiden sisältöelementtien sivut näyttävät, pitäisi käyttäjän pystyä määrittämään jonkinlaisella mekanismilla, esimerkiksi näitä tarkoituksia varten erityisesti tarkoitetussa sivupohjassa.

6.5.2. Sisältökatkelma (osio paletissa; engl. Content item)

Sisältökatkelma on sisällön perusyksikkö, käyttäjän muokattavissa oleva staattinen katkelma sisältöä. Sisältökatkelma on käytännössä HTML:ää, vaikka itse sisällönhallintajärjestelmän on hyvä pystyä tuottamaan sama sisältö muissakin formaateissa. Sisältökatkelmilla voi olla eri kenttiä, kuten ingressi, pääsisältö tai avainsanat, ja eri kenttiä voi asettaa joko näkyviksi tai näkymättömiin, esim. vain hakutoimintojen ja luokittelun käytettävissä olevaksi metadataksi.

Sisältökatkelman sisälle voidaan raahata muita sisältöelementtejä, ja määrätä ovatko ne sisältökatkelman sisällä tasattuina vasemmalle, keskelle vai oikealle. Toisin sanoen, käyttäjän ei tarvitse osata käyttää esiteltäviä asetteluja pystyäkseen sekoittamaan samalla sivulla eri sisältöelementtejä – asettelujen avulla positiota voi vain jakaa joustavammin ja toisaalta eri sivujen kesken yhtenäisesti

6.5.3. Mediasisältö (osio paletissa; engl. Media)

Mediasisältö esittää kuvat, Flash-animaatiot, videosisällön ja muut erilliset, WWW-sivulle upotettavissa tai linkitettävissä olevat tiedostomuodot. Järjestelmä tuottaa mediasisältöä sivulle raahatessa tarvittavan HTML:n, tai tiedostomuodosta riippuen vaihtoehtoisesti linkin väyläsivulle, joka linkittää kyseiseen mediaelementin tiedostoon.

Mikäli esimerkiksi käyttäjä haluaa laittaa PDF-tiedoston WWW-sivulle, Jacob Nielsenin suosituksen mukaisesti [Nielsen, 2003] tällaista väyläsivua käytetään, jotta käyttäjä tietää etukäteen, että on avaamassa PDF-tiedoston. Ominaisuus on pystyttävä myös kytkemään pois tarvittaessa, sallien tehdä suoran linkin PDF-tiedostoon käyttäjän harkinnan mukaisesti.

Nielsen tosin myös suosittelee, että PDF-tiedostojen käyttöä WWW:ssä vältettäisiin kaikin keinoin – ne sopivat kyllä tulostuskäyttöön, mutta niiden käytettävyys näytöltä katseltuina on heikko. Jää avoimeksi, millä tavalla järjestelmä voisi suositella käyttäjälle PDF-tiedoston WWW-muotoon muuttamista (mikäli mahdollista) ärsyttämättä käyttäjää. Nämä ovat esimerkkejä käyttäjää avustavista toiminnoista, johon viittaan Haasteet – luvun tavoitteessa 2.

6.5.4. Moduuli (osio paletissa; engl. module)

Moduulien tarkoitus on tehdä helpoksi muiden, tiettyihin erityisiin sisältötyyppeihin erikoistuneiden sisällönhallintaohjelmistojen tai -komponenttien saaminen saman sivuston saumattomiksi osiksi.

Moduuli (engl. module) on itsenäinen, jonkin sisällön (yleensä HTML-katkelman) tuottava ohjelmisto tai ohjelmakoodikatkelma. Tämä tutkielma kuvaa käyttöliittymän, eikä ota kantaa siihen, tuottaako järjestelmässä moduulin sisältönä näkyvän koodin esim. tietty luokka vai kokonainen, erillinen ohjelmisto. Moduuleja voi kirjoittaa myös laajennuksina sisällönhallintajärjestelmään. Moduulien yleisiä käyttötarkoituksia ovat esim. sivuston valikot, sivuston hakumoottori, kysely (engl. quiz, poll) tai vieraskirja. Moduuli käyttäytyy ehdotetussa käyttöliittymässä hyvin samankaltaisesti kuin mikä tahansa sisältökappale: moduulin raahaaminen valikkoon saa aikaan sivun, jossa kyseinen moduuli on pääsisältönä. Ero sisältökappaleisiin on se, että sisältökappaleen tuottama sisältö on staattista ja täysin käyttäjän muokattavissa, kun taas moduuli tuottaa sisältöä dynaamisesti ja se, miten käyttäjä voi moduulin tuottamaan sisältöön vaikuttaa, on moduulikohtaista.

Myös navigointiapuvälineet, kuten valikot ja sivustokartta ovat moduuleita, samoin kuin ns. *breadcrumb* (käyttäjän sijainti sivustolla yhdellä rivillä ilmaistuna, esimerkiksi: Maalariliike Oy -> Esittely -> Materiaalit). Sivustolla on yleensä yksi tai useampia valikoita, jotka näkyvät kaikilla sivuilla. Päävalikon rakenne määrää rakenteen sivuston käyttäjän näkökulmasta: kaikkien sivuston rakennetta kommunikoivien elementtien (breadcrumb, URL, ym.) pitäisi yleensä ottaen kommunikoida samaa rakennetta kuin valikko. Tähän on voitava tehdä poikkeuksia – yleinen esimerkki on se, että kaikkein tärkeimmät kohteet sivustolla sijoitetaan usein erilliseen valikkoon – ne ovat kuitenkin osa samaa hierarkiaa kuin päävalikon kohteet.

Valikko näkyy luonnollisesti myös järjestelmän muokkausnäkymässä ja sitä käyttäen sivustoa voi selailla. Tämän lisäksi valikkoon voi raahata paletista minkä tahansa elementin, joka luo uuden sivun tämä elementti pääsisältönä sisältöalueella.

6.6. Sivuston rakentaminen

Käyttäjän alettaessa rakentamaan uutta sivustoa häntä pyydetään valitsemaan sivustolleen jokin design (jonka voi tarvittaessa vaihtaa myöhemmin). Tämän jälkeen käyttäjä saa eteensä valitsemansa designin, tyhjän sivun jossa on valmiina valikko. Käyttäjä voi sitten raahata tälle tyhjälle, oletusarvoiselle etusivulle eri elementtejä. Uusi sivu luodaan raahaamalla paletista jokin elementti valikkoon, jolloin syntyy sivu ja vastaava kohta valikkoon. Eri elementit, raahattaessa valikkoon käyttäytyvät hieman eri tavoin:

- Sisältöelementti: Luo sivun, jolla on vain kyseinen sisältö.

- Mediasisältö: Sivu, jolla näkyy kyseinen mediasisältö, (kuva, Flash-animaatio, video, tms.)
- Asettelu: syntyy sivu, jossa on tyhjiä tiloja sisällölle, kyseisen asettelun mukaisesti.
- Design: Syntyy kyseistä designiä käyttävä sivu oletusasettelulla (ei jakoa, yksi positio).

7. Yhteenveto

Olen tässä tutkielmassa käsitellyt pienehköjen sivustojen sisällönhallinnan keskeisiä kysymyksiä, vertaillut kahta erilaista käyttöliittymä- ja käsitteistölähestymistapaa sekä esitellyt mallikäyttöliittymän, joka nähdäkseni vastaa moniin tämän päivän sisällönhallinnan opittavuuden ongelmiin.

Käyttöliittymän kokoamisessa olen pyrkinyt mahdollisuuksien mukaan sekä sisäiseen yhtenäisyyteen, että käyttämään eri käyttöliittymäelementtejä tuntemieni konventioiden mukaisesti. Kuitenkin lähestyessä todellisia käyttäjiä pitäisi tarkemmin ottaa selville, mitä tarpeita käyttäjillä todella on, ja johtaa sitten näistä käyttötapauksia. Esitettyä alkua käyttöliittymälle on vielä myöskin testattava todellisilla käyttäjillä, samoin kuin esitettyä käsitteistöä, joka näyttää kuitenkin alustavasti auttavan hallitsemaan eri puolia WWW-sivuston dynaamisuudesta melko kattavasti.

Tähän asti käyttöliittymä on suunniteltu esitellyn vertailun, kirjallisuuden ja henkilökohtaisen kokemuksen perusteella siitä, millä käsitteillä sisältöä pitäisi hallita: vastaava toiminnallisuus on käytössä ainakin osittain monissa jo olemassa olevissa järjestelmissä, mutta nähdäkseni ehdottamani käyttöliittymä on yksinkertaisempi ja suoravaikutteisuutensa ansiosta helpommin opittavissa. Lopullisen sovelluksen käytettävyys jää toki suurelta osalta riippumaan vähintäänkin vielä suunnittelematta olevista modaalisista dialogeista sekä rakenne- ja CSS-näkymistä. Myös W3C:n Authoring Tool Accessibility Guidelinesin [W3C, 2000a] noudattaminen ja käyttäjän tukeminen saavutettavien ja käytettävien sivustojen rakentamisessa on tärkeää tulevassa sisällönhallintajärjestelmän suunnittelutyössä.

8. Lähteet

[Mankoff et al., 2005] Jennifer Mankoff, Holly Fait, Tu Tran, Is your web page accessible?: a comparative study of methods for assessing web page accessibility for the blind, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '05. ACM Press, 41-50. Also available at http://doi.acm.org/10.1145/1054972.1054979

[Myers, 1991] Brad A. Myers 1991. Text formatting by demonstration, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Reaching

Through Technology. CHI '91. ACM Press, 251-256. Also available at http://doi.acm.org/10.1145/108844.108904

[Nielsen, 2003] Jacob Nielsen, Gateway Pages Prevent PDF Shock. Available at http://www.useit.com/alertbox/20030728.html (19.5.2006)

[Open Source Matters, 2006] Open Source Matters, Joomla! Administrator 1.0x Help. Available at http://help.joomla.org/content/section/16/153/ (19.5.2006)

[Robertson, 2002] James Robertson, How to evaluate a content management system. Available at http://www.steptwo.com.au/papers/kmc_evaluate/ (3.11.2006)

[Schneiderman, 1997] Ben Shneiderman, Direct manipulation for comprehensible, predictable and controllable user interfaces. Proceedings of the 2nd international Conference on intelligent User interfaces. IUI '97. ACM Press, 33-39. Also available at http://doi.acm.org/10.1145/238218.238281

[Sloan et al., 2000] David Sloan, Peter Gregor, Murray Rowan, Paul Booth, Accessible accessibility, Proceedings on the 2000 Conference on Universal Usability. CUU '00. ACM Press, 96-101. Also available at http://doi.acm.org/10.1145/355460.355480

[W3C, 1999] W3C, Web Content Accessibility Guidelines 1.0. Available at: http://www.w3.org/TR/WAI-WEBCONTENT/ (2006-03-11)

[W3C, 2000a] W3C, Authoring Tool Accessibility Guidelines 1.0. Available at: http://www.w3.org/TR/WAI-AUTOOLS/ (2006-03-11)

[W3C, 2000b] W3C, Techniques For Accessibility Evaluation And Repair Tools. Available at: http://www.w3.org/TR/AERT (2006-03-11)

[Wolber et al., 2002] David Wolber, Yingfeng Su, Yih Tsung Chiang, Designing dynamic web pages and persistence in the WYSIWYG interface. Proceedings of the 7th international Conference on intelligent User interfaces. IUI '02. ACM Press, 228-229. Also available at http://doi.acm.org/10.1145/502716.502770