Lappeenrannan teknillinen yliopisto
Teknistaloudellinen tiedekunta
Tietotekniikan osasto

TILASTOTIETOJEN GRAAFINEN ESITTÄMINEN WWW-SIVULLA

Tarkastaja: Professori Kari Smolander

Sampo Kettunen
Skinnarilankatu 28 C 8
53850 Lappeenranta
050 336 4655

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto
Tietotekniikan osasto

Sampo Kettunen

Tilastotietojen graafinen esittäminen WWW-sivulla

Kandidaatintyö

2007

35 sivua, 4 kuvaa, 4 taulukkoa ja 1 liite

Tarkastaja: Professori Kari Smolander

Hakusanat: WWW-sovellus, kehitystyökalu, avoin lähdekoodi
Keywords: web application, software engineering tool, open source software

World Wide Webin suosiolla on ollut merkittävä vaikutus yhteiskuntaan. WWW-sivut ovat helposti saatavilla ja sisällön tekeminen WWW:hen on helppoa. WWW-ympäristölle myös kehitetään paljon sovelluksia. WWW-sovellusten kehittämiselle ominaista on valinnanvapaus ja nopeuden tavoittelu.

WWW-sovellusten ohjelmoinnin mahdollistavat useat toisilleen vaihtoehtoiset tekniikat. Ne eroavat toisistaan suoritusnopeudessa, ominaisuuksien määrässä ja joustavuudessa. Ohjelmoinnissa käytetään apuna useita erilaisia menetelmiä. Apumenetelmiä ovat muun muassa työkalut ja valmiiden komponenttien hyödyntäminen. Valmiit komponentit voivat olla joko ilmaisia, avointa lähdekoodia tai maksullisia.

Tämän kandidaatintyön aikana valmistui sovellus, joka piirtää tilastotiedoista kaaviokuvia ja näyttää niitä dynaamisella WWW-sivulla. Sovellus pyrittiin toteuttamaan älykkäästi apumenetelmiä sopivasti hyödyntäen. Sovelluksen kehittämisessä käytettiin apuna sekä ohjelmointityökaluja että valmiita komponentteja. Kaaviokuvien tyypin ja ulkoasun haluttiin olevan käyttäjien muokattavissa. Toisaalta sovelluksen haluttiin olevan helposti laajennettavissa. Vaatimuksiin vastattiin tekemällä kaaviokuvien piirrosta osittain tietokannalla ohjelmoitava.

ABSTRACT

Lappeenranta University of Technology
Department of Information Technology

Sampo Kettunen

Representing Statistic Data on a Web Page

Thesis for the Degree of Bachelor of Science in Technology

2007

35 pages, 4 figures, 4 tables and 1 appendix

Examiner: Professor Kari Smolander

Keywords: web application, software engineering tool, open source software

The World Wide Web has had a significant impact on the society. Web pages are easily available and creating content for the web is easy. A lot of applications are also developed for the web. Freedom of choice and efforts to reach a fast pace are common features of web application development.

Programming web applications is possible by several alternative technologies. They differ from each other in execution speed, amount of features and flexibility. Many kinds of methods are used to help web application programming. Some of those methods are software engineering tools and the use of existing components. Existing components can be either completely free, open source or commercial.

A part of this thesis was to create an application which draws charts from statistical data and represents them on a dynamic web page. The application was made in a smart way while utilizing useful methods in a reasonable manner. Both software engineering tools and existing components were used in the development of the application. The types and appearance of the charts needed to be editable by the users. On the other hand, the application needed to be easily extendable. This was addressed by making the chart drawing partially programmable by a database.

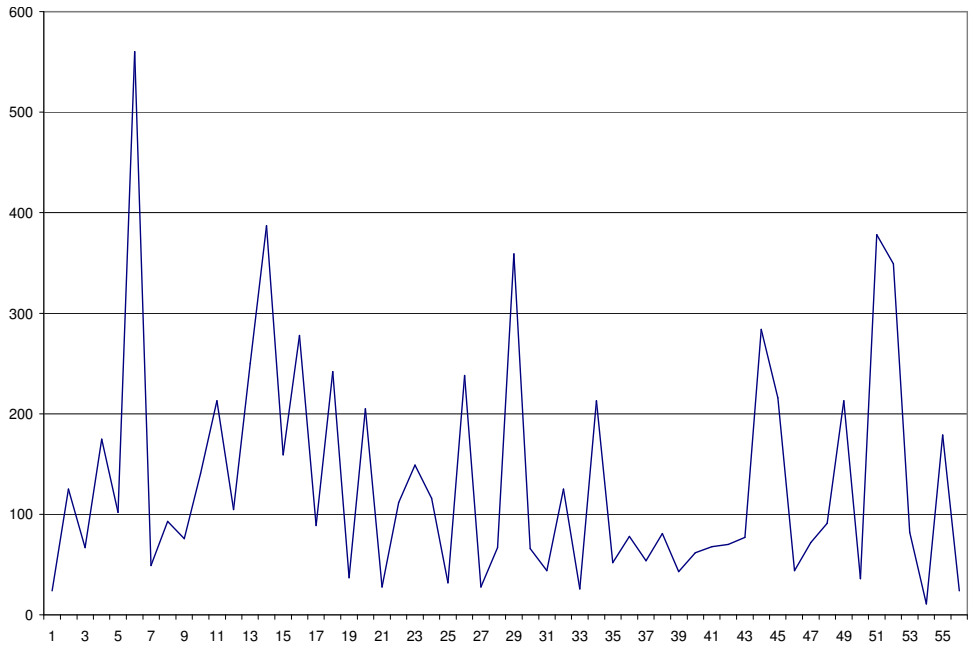
SISÄLLYS

1 JOHDANTO

World Wide Webin (WWW) suosion kasvamisella on ollut merkittävä vaikutus yhteiskuntaan. WWW:ssä on saatavilla valtava määrä tietoa, ja siellä käydään sähköistä liiketoimintaa. Myös WWW:n suosio sovellusten alustana on mittava. Syitä suosiolle on useita: WWW-sivuille voidaan sisällyttää kaikenlaista sisältöä, kuten tekstiä, kuvaa, ääntä ja videota. WWW-sivut ovat helposti saatavilla, koska niihin pääsee käsiksi eri puolilta maailmaa ja niiden lukeminen on mahdollista paitsi tietokoneilta myös muilta laitteilta, kuten puhelimilta, kämmenmikroilta ja pelikonsoleilta. Lisäksi sisällön tekeminen WWW:hen on suhteellisen helppoa, ja siihen on saatavilla lukuisia työkaluja.

WWW-sovellusten kehittäjien on hyödyllistä tietää alan kehityksestä. WWW-sovellusten kehittämiseen on olemassa useita vaihtoehtoisia ohjelmointitekniikoita. Tekniikoilla voidaan tehdä paljon samoja asioita, mutta kullakin tekniikalla on erityispiirteensä. Tekniikan valinta riippuu yleensä tilanteesta, mihin vaikuttavat sekä tekniikan soveltuvuus käyttökohteeseen että kehittäjien osaaminen. Valinnanvapauden lisäksi WWW-sovellusten kehittämiselle ominaista on nopeuden tavoittelu. Kehittämistä voidaan nopeuttaa hyvällä suunnittelulla, komponenttien uudelleenkäyttämisellä ja erilaisilla työkaluilla. Ilman näiden apumenetelmien tuntemusta kehittäjät tekevät liian paljon turhaa työtä.

Tämän kandidaatintyön tavoitteena oli suunnitella ja toteuttaa WWW-sovellus, jolla voidaan esittää tietokannassa olevia tilastotietoja. Työssä esitellään WWW-sovellusten ominaisuuksia, yleisiä ohjelmointitekniikoita sekä kehittämisessä käytettäviä apumenetelmiä. Tekniikoista ja menetelmistä valittiin sopivimmat työssä tehtävää sovellusta varten. Sovelluksen tehtävänä on piirtää tilastotiedoista havainnollisia kaaviokuvia. Kaaviokuva voi olla esimerkiksi viivadiagrammi, jollainen on nähtävissä kuvassa 1. Muita yleisiä kaaviokuvien tyyppejä ovat pylväs-, ympyrä- ja pistediagrammi.



Kuva 1. Esimerkki viivadiagrammista.

Työn toimeksiantajana toimi Stora Enso Oyj:n Pulp Competence Centre. Työssä tehty WWW-sovellus toimii siellä osana mittaustietokantaa. Dokumentti toimii paitsi apuna sovelluksen laajentamisessa myös kannustimena Stora Enson työntekijöille sovelluskehityksen apumenetelmien käyttöönotossa.

2 JOHDATUS WWW-SOVELLUKSIIN

WWW:lle on kahden vuosikymmenen aikana kehitetty tekniikoita, joilla on mahdollista tehdä monipuolisia sovelluksia. WWW-sovellukset voivat sijaita maailmanlaajuisessa Internetissä tai itsenäisissä sisäverkoissa, ja niitä käytetään WWW-selaimella. Yleisiä WWW-sovelluksia ovat sanomalehdet, uutiskirjeet, kirjat, pelit, verkkokaupat, pankkien verkkopalvelut, palveluiden tilauspalvelut, työympäristöt, keskustelut, yhteisöt, huutokaupat ja portaalit. Tässä luvussa kerrotaan WWW-sovellusten ominaisuuksista ja eroista perinteisiin sovelluksiin, toiminnallisuuden ohjelmoinnista sekä asiakas- ja palvelinpuolen tekniikoista.

2.1 Ominaisuudet

WWW-sovellus on erityistapaus asiakas-palvelin-arkkitehtuurista, joka on yleinen tapa hajauttaa sovellus. Asiakas-palvelin-arkkitehtuurissa sovelluksen toiminnallisuus on jaettu osiin, asiakaspuolelle ja palvelinpuolelle. Sovelluksen resurssit sijaitsevat palvelinpuolella ja niitä käytetään asiakaspuolelta. Resurssien käyttö vaatii pyynnön lähettämistä palvelinpuolelle ja vastauksen vastaanottamista. Useat asiakkaat voivat käyttää samoja resursseja. Asiakas-palvelin-arkkitehtuurin sovellus on usein kolmitasoinen. Tasot ovat käyttöliittymä, väliohjelmisto ja tietokanta. Käyttöliittymä tarjoaa rajapinnan ihmisen ja tietokoneen välille. Se tuottaa yleensä graafista tietoa ja kerää käyttäjän syötteitä. WWW-sovelluksissa käyttöliittymänä toimii WWW-selain. Väliohjelmisto hallitsee pyyntöjen suorittamista taaten, että ne noudattavat sääntöjä. Tietokannassa sijaitsee varsinainen data. Tasojen erottamisen etuina on, että kukin taso on itsenäinen voiden muuttua ilman muiden tasojen muuttamista ja että asiakaspuoli on eristettynä sovelluksen resursseista. [1, 2, 3]

WWW-sovellukset eroavat perinteisistä työpöytäsovelluksista asennuksen ja viestinnän luonteessa. Koska asiakaspuolen ohjelmistona toimii selain, sovellusta voidaan käyttää alustasta riippumattomasti, mikä tekee suuremman käyttäjäkunnan mahdolliseksi. Lisäksi

sovelluksen muuttaminen vaatii toimenpiteitä ainoastaan palvelinpuolella. WWW-sovelluksen viestintä tapahtuu pääosin yhteydettömällä HTTP-protokollalla, jonka tarkoitus ei ole olla tehokas vaan monipuolinen ja virheitä sietävä. Asiakaspuolen ja palvelinpuolen välinen kommunikointi voi yksinkertaisesti perustua pelkkään WWW-sivujen hakemiseen. [3, 4]

Myös sovelluskehitys WWW:ssä eroaa työpöytäsovellusten kehityksestä. Usein sivujen ulkoasun suunnitteluun käytetään paljon huomiota ja kehitykseen kuuluu myös sisällön kehittämistä. Käytettävyyden suunnittelu on monimutkaista, koska käyttäjien taustat voivat olla hyvin erilaisia. Monesti WWW-pohjaiset järjestelmät täytyy tehdä melko nopeasti, jolloin suunnittelussa ja testauksessa ei voida olla kovin perusteellisia. Lisäksi WWW-pohjaisten järjestelmien kehittäjillä on hyvin erilaisia taustoja, sillä sovelluskehitykseen WWW:ssä liittyy useita aihealueita: ohjelmistotuotanto, hyperteksti, tietojenkäsittely, vaatimusmäärittely, järjestelmäsuunnittelu, arkkitehtuurisuunnittelu, projektinhallinta, testaus, ihmisen ja tietokoneen vuorovaikutus sekä multimedia. [5]

2.2 Toiminnallisuuden ohjelmointi

HyperText Markup Language (HTML) on WWW-sivujen tavallisin tekniikka. HTML-sivut ovat tekstiä ja elementtejä sisältäviä staattisia sivuja. WWW-sivuille on saatu aikaan useampia käyttötarkoituksia tekemällä tekniikoita, joilla HTML-sivujen yhteydessä voidaan käyttää ohjelmointia. Tekniikat, joilla toiminnallisuutta ohjelmoidaan, jaotellaan tyypillisesti asiakaspuolen ja palvelinpuolen ohjelmointiin. Sekä asiakaspuolen että palvelinpuolen ohjelmointiin on olemassa useita vaihtoehtoisia tekniikoita. [6, 7]

Palvelinpuolen ohjelmoinnin tärkein ominaisuus on mahdollisuus tuottaa räätälöityjä WWW-sivuja ohjelmallisesti, mikä on edellytys WWW-sovellusten toimimiselle. Tekniikoiden avulla voidaan esimerkiksi luoda vastauksia käyttäjän tarpeiden, oikeuksien tai kyselyjen mukaan. Käyttäjän näkökulmasta sivu on palvelinpuolen ohjelmoinnista huolimatta HTML-kielellä

määritelty kokonaisuus sisältöä ja muotoiluja. Kun sivua pyydetään, palvelimella suoritettava ohjelma luo lähetettävän sivun dynaamisine sisältöineen. Palvelinpuolen tekniikat mahdollistavat verkkopalvelujen tekemisen, mutta niillä yksistään ei voi tehdä käytettävyydeltään kovin hyviä WWW-sovelluksia. Tällaisissa kevyen asiakkaan WWW-sovelluksissa käyttäjä saattaa esimerkiksi joutua käymään läpi useita sivuja suorittaakseen vain yhden asian tai käyttäjän on rajattava tietoja monimutkaisilla lomakkeilla. Lisäksi vuorovaikutus tapahtuu aina sivuja vaihtamalla, jolloin käyttäjä joutuu odottamaan uuden sivun latautumista. [8, 9]

Asiakaspuolen ohjelmoinnilla HTML-sivuille voidaan sisällyttää toiminnallisuutta, joka mahdollistaa paremman käytettävyyden. Asiakaspuolen tekniikat eivät korvaa HTML-kieltä, vaan tarjoavat HTML-sivuille lisäominaisuuksia. Asiakaspuolen tekniikat suoritetaan käyttäjän koneella, ja niiden tärkein ominaisuus on mahdollisuus reagoida käyttäjän tekoihin ottamatta yhteyttä palvelimelle. Asiakaspuolen tekniikoilla käyttäjälle voidaan antaa palautetta lataamatta uusia WWW-sivuja. Joissain tapauksissa käyttäjän on tarpeellista saada jatkaa sivun käyttämistä, vaikka selain odottaa vastausta palvelimelta. Asiakaspuolen tekniikoiden edistyessä WWW-sovelluksista on mahdollista tehdä yhä enemmän työpöytäsovellusten kaltaisia. [8, 9]

WWW-sovelluksen työtaakkoja voidaan jakaa asiakaspuolelle ja palvelinpuolelle tilanteen mukaan. Kevyen asiakkaan ja raskaan palvelimen tapauksessa asiakaspuoli on ainoastaan vastuussa käyttöliittymästä ja palvelin on vastuussa suurimmasta osasta toimintoja. Toisen ääripään, raskaan asiakkaan ja kevyen palvelimen, tapauksessa palvelinpuoli on vastuussa ainoastaan tietokannan hallinnasta ja asiakaspuoli on vastuussa suurimmasta osasta toimintoja. Esimerkiksi käyttäjän syötteen tarkistaminen voidaan suorittaa joko palvelinpuolella tai asiakaspuolella. Työtaakkojen jakamisesta on parasta päättää sovelluksen suunnitteluvaiheessa. [1]

Asiakaspuolen ollessa kevyt, asiakaskone viestii paljon palvelinkoneen kanssa ja palvelin suorittaa lähes kaiken prosessoinnin. Asiakaskoneen huoltamisesta ei tarvitse huolehtia kovin usein, sillä sen laitteistolta ei ole suuria vaatimuksia. Äärimmäisessä tapauksessa asiakaspuolen tarvitsee pystyä vain lukemaan HTML-sivuja. JavaScriptiä käytettäessä asiakaspuolen ajatellaan olevan myös kevyt. Uudet käyttäjät voivat helposti aloittaa sovelluksen käyttämisen, sillä asennuksia ei tarvita. Data on keskitetty, jolloin sen yhdenmukaisuudesta ja täydellisyydestä voidaan olla varmoja. Lisäksi ohjelmisto voidaan päivittää ja saada käyttöön välittömästi kaikille käyttäjille. [4]

Asiakaspuolen ollessa raskas, palvelinpuolelta kopioitua tietoa säilytetään asiakaspuolella ja asiakaskone suorittaa suuren osan prosessoinnista. Tämä mahdollistaa tiedon nopean uudelleenkäyttämisen: tietoa ei tarvitse hakea palvelinpuolelta asti, mikä mahdollistaa paremmat vasteajat. Toimintojen suorittaminen asiakaspuolella voi olla aiheellista myös silloin, kun palvelulla on paljon käyttäjiä ja palvelimella on paljon kuormaa. Toisaalta raskaan asiakasohjelman lataaminen kestää pitkään. Lisäksi suunnittelijoiden on otettava huomioon, että ohjelmalisäkkeiden asennus voi olla estetty asiakaskoneilla tietoturvan vuoksi. Raskaat asiakkaat vaativat Java- tai Flash-yhteensopivuuden. [4]

2.3 Palvelinpuolen tekniikat

Alun perin dynaamiset WWW-sivut saatiin aikaan Common Gateway Interfacen (CGI) avulla. CGI on keino, jolla palvelin voi ohjata sivupyynnön jollekin palvelimella olevalle ohjelmalle. Näitä ulkopuolisia ohjelmia on kirjoitettu perinteisillä ohjelmointikielillä, kuten C-kielellä, mutta merkkijonojen muokkaamiseen keskittyneet kielet, kuten Perl, ovat olleet suositumpia. Perl kehitettiin jo vuonna 1987. Kieli on hyvin kehittynyt, sillä on paljon käyttäjiä ja se on saatavissa kaikille suosituimmille käyttöjärjestelmille. Kieltä ei kuitenkaan kehitetty WWW-sovelluksia varten, mistä johtuen se on WWW-käytössä hidas ja liian monimutkainen. [10]

Microsoftin ASP-tekniikan avulla komentosarjakielillä, kuten JavaScript ja VBScript, kirjoitettuun koodiin voidaan liittää ohjelmistokomponentteja, joita käsitellään kuin olioita. ASP on helppo oppia, riittävä useimmille WWW-sovelluksille sekä melko tehokas. ASP on tehokkain Windows Server 2000 -käyttöjärjestelmän ja Microsoftin WWW-palvelimen kanssa. ASP:n käyttö siis maksaa, mutta tukea on hyvin saatavissa. ASP.NET on ASP:sta kehitetty Microsoftin uudempi tekniikka. [10]

ColdFusion on Adoben omistama merkintäkieli. Suurin osa ColdFusionin käytöstä perustuu olemassa olevien tunnisteiden käyttöön, mutta myös räätälöityjen tunnisteiden ohjelmointi on mahdollista perinteisillä ohjelmointikielillä. ColdFusion on mainio kieli ihmisille, jotka eivät osaa ohjelmoida, sillä jopa tietokannan käyttö on mahdollista olemassa olevien tunnisteiden avulla. ColdFusion-palvelinohjelmisto maksaa, mutta toisin kuin ASP se on saatavissa useille alustoille. [10]

PHP on avoimen lähdekoodin tekniikka, jonka ajatellaan syrjäyttäneen Perlin. PHP on tehokas ja saatavissa useimmille alustoille. Linux-pohjainen PHP:ta käyttävä WWW-palvelin on täysin ilmainen ja hyvin suosittu ohjelmistokokonaisuus. PHP-kielen syntaksi on perinteisten ohjelmointikielten kaltainen. Ohjelmointia osaaville PHP on helposti lähestyttävä, mutta muille sen opettelu on hieman työläämpää. [10]

Java on täysiverinen ohjelmointikieli. Sillä voidaan tehdä hienostuneita ohjelmia, mutta sen opettelu ei ole helppoa. Varsinkin aloittelevan ohjelmoijan on huomattavasti nopeampaa saada aikaan pieniä WWW-sovelluksia komentosarjakielillä. Javan suurimpia vahvuuksia kilpailevaan ASP.NET-tekniikkaan verrattuna on sillä kirjoitetun koodin alustariippumattomuus. Kerran kirjoitettua koodia voi ajaa missä tahansa. [10]

2.4 Asiakaspuolen tekniikat

Netscape ja Sun julkaisivat JavaScriptin vuonna 1995. JavaScript-komentosarjat ladataan WWW-sivun lataamisen yhteydessä ja suoritetaan asiakkaan koneella. Komentosarjat vuorovaikuttavat HTML-sivun kanssa. Suosituimmat selaimet tukevat JavaScriptiä, koska se on alustasta riippumaton. JavaScriptin käyttö tietokoneella voidaan kuitenkin tietoturvasyistä estää. Ajax-menetelmällä JavaScript-komentosarja voi tehdä HTTP-kyselyjä, jotka eivät näy käyttäjälle sivulatauksina. Tällä keinolla sivulle voidaan hakea uutta tietoa nopeasti, sillä siirtää tarvitsee vain tarpeellinen määrä tietoa. [11]

Flash- ja Java-sovelmat ovat raskaan asiakaspuolen tekniikoita. Sovelmat ovat HTML-sivuille upotettavia ohjelmia, jotka siirtyvät sivua ladattaessa asiakaskoneelle. Sovelmat luovat omat graafiset käyttöliittymänsä, ja niillä voidaan tehdä esimerkiksi pelejä. Sovelmien suorittaminen vaatii ohjelmalisäkkeiden asentamisen selaimeen. Sekä Java, että Flash mahdollistavat äänen ja videon soittamisen selaimessa, mikä ei ole JavaScriptillä vielä mahdollista. [8, 11]

3 APUMENETELMÄT WWW-SOVELLUSTEN KEHITTÄMISESSÄ

Samalla kun WWW-sovellukset ovat yleistyneet nopeasti, niistä on tullut yhä monimutkaisempia. WWW-sovellusten kehittämisen vaaditaan jatkuvasti olevan yhä nopeampaa ja halvempaa. Silti sovellusten tulisi olla lähes virheettömiä. Tämän takia ammattilaiset ovat pyrkineet tehostamaan sovelluskehitystään erilaisilla menetelmillä. WWW-sovellusten kehittämisen apuna voidaan käyttää samoja menetelmiä kuin perinteisten sovellusten apuna, mutta joitakin menetelmiä on jalostettu nimenomaan WWW-ympäristöön soveltuviksi. Tässä luvussa kerrotaan suosituista apumenetelmistä, joita ovat suunnittelussa käytettävät kuvauskielet, erilaiset kehitystyökalut ja avoin lähdekoodi.

3.1 Kuvauskielet suunnittelussa

WWW-sovelluksia tehdään usein ilman kunnollista suunnittelua. Tyypillinen tapaus on, että WWW-sovellusta rakennetaan hiljalleen ominaisuuksia lisäämällä samalla kun sovellus on käytössä. Tämä on johtanut siihen, että monet tärkeät laadulliset seikat jäävät usein vaille huomiota sovellusten kehittäjiltä. Näitä seikkoja ovat esimerkiksi navigoitavuus, helppokäyttöisyys, yhteensopivuus, tietoturva, luettavuus ja toimintavarmuus. [5]

Ongelmien välttämiseksi WWW-sovelluksia varten on täytynyt kehittää kuvauskieliä, jotka ovat tarkempia kuin perinteisille sovelluksille sopivat kuvauskielet. Kuvauskielillä sovellukset suunnitellaan korkealla abstraktiotasolla, ja useissa tapauksissa lopullinen sovellus voidaan generoida abstraktiomäärittelyistä. Aluksi kuvauskielet olivat riittämättömiä määrittelemään tarpeeksi monimutkaisia sovelluksia, koska ne soveltuivat vain vanhanaikaisten, useista sivuista koostuvien, WWW-sovellusten suunnitteluun. Sovelmia sisältäville WWW-sovelluksille on täytynyt kehittää erilaisia kuvauskieliä. [9]

3.2 Kehitystyökalut

Kehitystyökalujen käyttö nopeuttaa sovelluskehitystä. Kehitystyökaluja sovelluskehityksessä on käytetty ensimmäisistä kääntäjistä lähtien. Aluksi työkalut ainoastaan tarjosivat mahdollisuuden kehittää ohjelmia, mutta myöhemmin työkaluihin on kehitetty ominaisuuksia, jotka auttavat käyttäjää automatisoimalla yleisiä tehtäviä. Erilaisten työkalujen määrä ja kirjavuus on lisääntynyt valtavasti ajan saatossa ohjelmien monimutkaistuessa. [12]

Perinteisiä kehitystyökaluja ovat kääntäjien lisäksi muiden muassa tekstieditorit ja virheitä paikantavat työkalut; jotkut työkalut auttavat keräämään vaatimuksia, suunnittelemaan käyttöliittymiä, luomaan kyselyjä, määrittelemään viestejä, suunnittelemaan arkkitehtuureja, testaamaan ohjelmia tai joko hallitsemaan versioita, asetuksia tai tietokantoja. Suuret ohjelmointiympäristöt sisältävät useita kehitystyökaluja, jolloin työkalut voivat vuorovaikuttaa toistensa kanssa. [12]

3.3 Avoimen lähdekoodin hyödyntäminen

Ohjelmistoyritykset ovat jo pitkään halunneet kilpailuetunsa säilyttämiseksi pitää lähdekoodinsa oikeudet itsellään, jolloin lähdekoodin sanotaan olevan suljettua. Ohjelmoinnin harrastajat ovat kuitenkin huomanneet uudelleen käytettävyyden hyödyllisyyden ja halunneet julkaista lähdekoodejaan. Avoimen lähdekoodin projekteja on olemassa tuhansia ja niiden merkitys kasvaa nopeasti. Avoimen lähdekoodin käytöstä voi olla rahallista hyötyä; sen ympärille on syntynyt jopa jonkin verran liiketoimintaa. [13]

Open Source Initiative -organisaatio ylläpitää listaa niistä kymmenistä lisensseistä, joita käyttäviä tuotteita saa kutsua avoimen lähdekoodin tuotteiksi [14]. Lisenssien avulla tekijänoikeuden haltija voi myöntää käyttöoikeuksia tuotteelleen, mutta ne myös velvoittavat noudattamaan lisenssiehtoja. Vaikka tuote on helposti ladattavissa, ehdot jokaisen elementin, version ja käyttötarkoituksen osalta täytyy selvittää. Mikäli lisenssiehtoja ei noudata, voi

joutua oikeuteen juridisten velvoitteiden rikkomisesta. Lisenssit eroavat sisällöltään hieman toisistaan, mutta niiden kaikkien on täytettävä tietyt Open Source Initiativen määrittelemät levitysehdot. Useimmat lisenssit sanelevat, että avoimesta lähdekoodista johdettujen ohjelmien on oltava myös saman lisenssin alaisia, mutta jotkut lisenssit sallivat jopa johdannaisten kaupallistamisen. [13]

Avoimesta lähdekoodista voi olla paljon hyötyä sovelluskehityksessä. Avoin lähdekoodi voidaan jakaa kolmeen kategoriaan: sovelluskehitystä helpottavat kehitysympäristöt, toimintaympäristönä toimivat ajonaikaiset alustat sekä muihin ohjelmiin liitettävät komponentit. Kehitysympäristöjä ja ajonaikaisia alustoja on olemassa myös kaupallisina, mutta ohjelmiin liitettävät lisäkomponentit on yleensä joko tehtävä itse tai on käytettävä avoimen lähdekoodin komponentteja. Kaikkia avoimen lähdekoodin käyttäjiä ei kiinnosta varsinaisesti ideologia tai avoimuus vaan avoimuuden tuomat kustannussäästöt. Monet avoimen lähdekoodin ohjelmistot ovat yhtä laadukkaita kuin vastaavat kaupalliset ohjelmistot. Toisaalta kustannuksia tulee muustakin kuin investoinneista lisensseihin, avoimen lähdekoodin ohjelmien lisenssien ehtoja on aina noudatettava eivätkä ohjelmien tekijät yleensä sitoudu antamaan tuotteilleen minkäänlaista tukea. Avoimen lähdekoodin ohjelmia käytettäessä on tapauskohtaisesti selvitettävä, onko niiden käyttö kannattavaa. [13]

Tehokkuuden ja edullisuuden lisäksi laatu on ratkaiseva tekijä valmiiden tuotteiden valinnassa. Avoimen lähdekoodin väitetään olevan suljettua lähdekoodia laadukkaampaa, koska avointa lähdekoodia voi lukea kuka tahansa. Jotkut ovat kuitenkin väittäneet päinvastaista [15]. Lukuisten avoimen lähdekoodin tuotteiden joukosta on varminta valita ne, jotka ovat osoittautuneet luotettaviksi ja hioutuneet käyttökelpoisimmiksi. On helpointa luottaa tunnettuihin ja kypsyneisiin ohjelmiin, joiden käytöstä on paljon kokemusta, kuin pieniin ja tuoreisiin tuotteisiin. [13]

Suurin juridinen riski avoimen lähdekoodin tuotteiden käytössä on mahdollinen kolmansien osapuolien immateriaalioikeuksien loukkaaminen, sillä lähdekoodin alkuperän selvittäminen

on hankalaa. Jos ohjelmaan on tarkoituksella tai vahingossa päässyt laittomia osia kolmannen tahon suojatusta koodista, väärinkäytöksestä ei pelkästään ole vastuussa avoimen lähdekoodin tekijät vaan kaikki ne henkilöt, jotka käyttävät, jakelevat ja muokkaavat kyseistä ohjelmaa. Epävarmoissa tapauksissa on parasta lopettaa tuotteen käyttö ja siirtyä käyttämään korvaavaa tuotetta. Pienten tuotteiden tapauksessa on myös mahdollista ohjelmoida vastaavat tuotteet itse, jolloin tekijänoikeudet ovat varmasti kunnossa. [13]

4 TILASTOTIETOJA ESITTÄVÄN SOVELLUKSEN MÄÄRITTELY JA SUUNNITTELU

Työn aikana tehtiin WWW-sovellus, jonka avulla voidaan tutkia tietokantaan tallennettuja tilastotietoja. Tässä luvussa esitellään sovellukselle asetetut vaatimukset, aikaisemmin käytössä ollut sovellus sekä suunnitelma uudesta sovelluksesta.

4.1 Vaatimukset

Sovelluksen on luonnollisesti sovittava siihen tehtävään, johon se on tarkoitettu. Yleiset käytettävyysvaatimukset ovat sovelluksen opittavuus, tehokkuus ja miellyttävyys. Tätä varten on tiedettävä ketkä sovellusta käyttävät, mikä on käyttäjän tavoite, missä sovellusta käytetään, mitä käyttäjät ovat tekemässä käyttäessään sovellusta ja mitä vaatimuksia näistä seuraa tuotteen käytettävyydelle. [16]

Tyypillinen sovelluksen käyttäjä haluaa selata tietokannan tietoja tehdäkseen niistä johtopäätöksiä, joista on hyötyä jossain kyseisiin tietoihin liittyvässä asiassa. Valtavan tietomäärän takia käyttäjä haluaa yleensä tarkastella kerrallaan vain osaa tiedoista. Kaaviokuvat ovat hyvä tapa abstrahoida suuri määrä tietoa, mutta joskus käyttäjän on tarpeellista saada tietää tarkkoja yksittäisiä arvoja. On aiheellista pystyä muokkaamaan kaaviokuvan tekstejä ja muita ominaisuuksia niin, että se olisi mahdollisimman selkeä. On myös aiheellista pystyä tallentamaan kaaviokuvia sekä asetuksia myöhempää käyttöä varten.

Sovelluksen käyttäjäkunta ei ole tarkasti tiedossa. Voidaan olettaa, että sovelluksen käyttäjät osaavat tietokoneiden käytön perusteet. Sovelluksen täytyy olla niin yksinkertainen tai opastaa käyttäjää niin hyvin, ettei sen käyttö vaadi koulutusta. Sovellusta käytetään eri puolilta maailmaa, joten keskitettyjen mittaustietojen on oltava helposti saatavissa Internetin kautta. Sovelluksen tulee luonnollisesti pystyä palvelemaan useita samanaikaisia käyttäjiä. Koska

sovellusta käyttävien tietokoneiden käyttöjärjestelmiä ja asetuksia ei tunneta tarkasti, sovellus ei saa vaatia asennusta.

4.2 Aikaisemmin käytössä ollut sovellus

Tilastotietojen tarkastelua varten oli jo aikaisemmin ollut käytössä WWW-sovellus. Sovelluksella käyttäjä oli voinut selata mitattuja tilastotietoja. Ensin käyttäjän oli täytynyt rajata tiedot lomakkeella. Käytännössä lomakkeessa oli ollut valittavana tietojen lähde, mitattujen arvojen tyyppi, aikaväli, tilastollisia muuttujia, tietojen järjestys sekä tietojen tyyppien suodattimia. Rajaamisen jälkeen käyttäjä oli saanut näkyvilleen sivun, jolla oli ollut kyseisen lomakkeen lisäksi tietojoukosta luotu kaaviokuva. Käyttäjä oli voinut rajata kaaviokuvasta alueen, josta oli voitu piirtää uusi kaaviokuva lataamalla sivu uudestaan. Kaaviokuvien muodostuksessa käytetyt tietokantahaut olivat tallennettavissa tietokantaan, joten samat kaaviokuvat oli pystytty palauttamaan myöhemmin pelkän tunnusnumeron avulla.

Sovelluksen palvelinpuolen tekniikoina olivat olleet PHP ja MySQL. Tietojen rajaukseen käytettävä lomake oli luotu aina PHP:n avulla MySQL-tietokannan datan mukaan. Lomakkeeseen syötetyistä tiedoista oli luotavissa SQL-kysely, joka tallennettiin tietokantaan. Kuvan muodostaminen tietojoukon perusteella oli tapahtunut PHP-kielisen JpGraph-kirjaston avulla. JpGraph esitellään myöhemmin. Asiakaspuolella sivut olivat näyttäneet tavallisilta HTML-sivuilta ja kaaviokuvat bittikarttakuvilta. Alueen valitseminen hiirellä kuvasta oli ollut mahdollista JavaScript-komentosarjojen avulla.

Aikaisemmin käytössä ollut sovellus ei ollut vastannut käyttäjien tarpeita. Eräs häiritsevä ongelma olivat olleet sivulataukset, jotka olivat haitanneet sovelluksen miellyttävyyttä. Sivun päivittäminen oli aiheuttanut sivun vierityksen nousemisen sivun ylälaitaan ja kaaviokuvien katoamisen hetkeksi. Kommunikaatio asiakaspuolen ja palvelinpuolen välillä kaaviokuvia muodostettaessa oli ollut välttämätöntä, sillä kaaviokuvat muodostettiin palvelinpuolella.

Sovelluksen suurin puute oli ollut, etteivät käyttäjät voineet vaikuttaa tarpeeksi kaaviokuvien ominaisuuksiin. Kaaviokuva oli ollut aina viivadiagrammi, vaikka joissakin tapauksissa toisenlainen kaaviokuva olisi ollut parempi. Uuden sovelluksen haluttiin pystyvän piirtämään käyttäjän valinnan mukaan ainakin viivadiagrammeja, pylväsdiagrammeja ja pistediagrammeja. Kaaviokuvien selkeyteen oli vaikuttanut myös se, ettei niiden teksteihin voinut vaikuttaa. Käyttäjän haluttiin pystyvän muokkaamaan sekä kaaviokuvien tekstejä että värejä. Myös tilastollisia muuttujia haluttiin näytettävän kaaviokuvissa. Asetukset tuli pystyä tallentamaan haluttaessa tietokantaan myöhempää käyttöä varten.

4.3 Arkkitehtuurisuunnittelu

Koska käytössä oli jo aiemmin ollut sovellus, uudesta sovelluksesta kannatti tehdä samankaltainen. Uuden sovelluksen oli ratkaistava ainoastaan aiemmassa toteutuksessa todetut ongelmat. Käyttäjien kannalta helpointa oli, että he joutuivat opettelemaan vain vähän uusia asioita. Arkkitehtuurin valinnassa täytyi ottaa myös huomioon, etteivät kaikki tekniikat olleet käytettävissä. Raskaat asiakasohjelmat vaativat tukea asiakaspuolelta, mutta palvelimella käytettävät tekniikat puolestaan vaativat asennuksen palvelimelle sekä joissain tapauksissa lisenssien ostamisen. Koska tehtävä sovellus oli pieni, väliohjelmiston tekniikkana päätettiin käyttää jo palvelimelle asennettua PHP:ta. Näin ollen aiemmasta toteutuksesta voitiin säilyttää lomake, jolla tietoja rajattiin. Myös aikaisemmassa sovelluksessa käytössä ollut MySQL-tietokanta säilytettiin uuteen sovellukseen.

Aluksi kaaviokuvat suunniteltiin piirrettävän Java-sovelmassa. Väliohjelmisto olisi vastannut ainoastaan tiedon välittämisestä tietokannasta sovelmalle. Sovelma olisi säilyttänyt tietokannasta haettua dataa asiakaspuolella niin kauan kuin sivu olisi ollut auki. Kaaviokuvia olisi voitu piirtää ilman kommunikointia palvelimen kanssa silloin, kun piirrettävän näkymän data olisi jo ollut sovelman muistissa. Tästä johtuen kaaviokuvien piirtäminen olisi ollut nopeaa eikä se olisi kuormittanut palvelinta. Käyttäjän ei olisi tarvinnut odottaa datan

saapumista muuttaessaan kaaviokuvien näkymää, sillä uusia näkymiä olisi voitu aina piirtää välittömästi ilman niitä pisteitä, jotka eivät ehtineet saapua palvelimelta. Uusien pisteiden koordinaattien saapuessa näkymä olisi piirretty uudelleen täydellisenä. Javan käytöstä luovuttiin, kun kävi ilmi, ettei kaikilla koneilla, joilta sovellusta käytettäisiin, ollut oikeuksia asentaa ja ajaa Java-sovelmia. Kaaviokuvat päätettiin piirrettävän väliohjelmistossa kuten aiemmassakin sovelluksessa.

Asiakaspuoli tuli vastaamaan käyttöliittymän esittämisestä, sivupyyntöjen lähettämisestä ja kuvien rajaamisesta. Palvelinpuoli tuli vastaamaan tietokannan käyttämisestä, kaaviokuvien piirtämisestä ja luonnollisesti sivujen luomisesta. Käyttäjien vaatimus kyetä muuttamaan kaaviokuvien muotoiluja ja tekstejä päätettiin ratkaista tekemällä tietokantaan tauluja, joilla kaaviokuvien piirtoa voitaisiin ohjelmoida. Kaaviokuvan päivittämisestä koituvat ongelmat päätettiin ratkaista JavaScriptin ominaisuuksia hyödyntäen. Myös kuvien rajaamiseen päätettiin käyttää JavaScript-komentosarjoja.

Sovelluksen käyttöliittymä päätettiin koostaa kahdesta WWW-sivusta, jotka ovat kaaviokuvan esittävä sivu ja muotoilujen muokkaamisen mahdollistava sivu. Kaaviokuvan esittävällä sivulla käyttäjä voi määritellä lomakkeella rajausehdot tilastotiedoille ja nähdä kaaviokuvan. Muotoilujen muokkaamisen mahdollistavalla sivulla käyttäjälle luodaan valikot sen mukaan, mitä vaihtoehtoja tietokannassa on. Valikoissa oli kuitenkin kiinnitettävä huomiota sovelluksen käytettävyyteen, sillä liian monimutkaiset valikot haittaavat käyttäjien toimimista. Osan valikoista huomattiin olevan piilotettavissa riippuen toisten valikoiden valituista asetuksista. Sivun osien päätettiin muuttuvan käytön aikana Ajax-tekniikan avulla.

5 SOVELLUKSEN TOTEUTTAMINEN

Tässä luvussa kuvaillaan kuinka edellisessä luvussa suunniteltu sovellus toteutettiin. Sovelluksen toteuttamisessa käytettiin apuna työkaluja ja valmiita komponentteja, jotka esitellään tässä luvussa. Tietokannan taulut, joilla kaaviokuvien muotoilut ovat ohjelmoitavissa, esitellään myös tässä luvussa. Lisäksi tässä luvussa esitellään kaaviokuvia esittävän ja muotoilujen muokkaamisen mahdollistavan sivun rakenteet.

5.1 Käytetyt työkalut

Sovellus kehitettiin Windows XP -ympäristössä. Sovelluksen lähdekoodit kirjoitettiin ilmaista avoimen lähdekoodin PSPad-tekstinkäsittelyohjelmaa käyttäen. PSPad on pienikokoinen tekstinkäsittelyohjelma, joka ei vaadi asennusta. Ohjelma on yksi useista vastaavista tekstinkäsittelyohjelmista, jotka auttavat ominaisuuksillaan käyttäjää tiettyjen kielten ohjelmoinnissa. PSPad kykenee korostamaan väreillä ja tekstin muotoiluilla muun muassa PHP-komentosarjoja, JavaScript-komentosarjoja, HTML-dokumentteja ja SQL-lauseita.

Sovellusta testattiin kehittämisen aikana Ubuntu Linux -ympäristössä. Ympäristöön oli asennettu Apache 2.2.3 -WWW-palvelin, PHP 5.2.1 -tulkki ja MySQL 5.0.38 -tietokanta. Testauksessa sovellusta käytettiin Windows XP -ympäristöstä, jossa WWW-selaimena käytettiin sekä Microsoft Internet Explorer 6.0:a että Mozilla Firefox 5.0:a.

5.2 Käytetyt valmiit komponentit

Sovelluksen toteutuksessa käytettiin useita valmiita komponentteja, mikä vähensi työmäärää. Luvat komponenttien käyttämiseen varmistettiin: komponenttien lisenssit joko ostettiin tai ne antoivat ilmaiseksi luvan rahallisen hyödyn tavoittelemiseen. Ajax-kutsujen lähettämistä päätettiin helpottaa käyttämällä Prototype-kehystä. Kaaviokuvien rajaamiseen päätettiin

käyttää JavaScript Image Cropper UI -kirjastoa. Kaaviokuvien piirtämiseen päätettiin käyttää JpGraph-kirjastoa. Tässä luvussa esitellään kyseiset komponentit.

5.2.1 Prototype

Prototype-kehys laajentaa JavaScript-kieltä tarjoamalla ohjelmoijien käyttöön yleiskäyttöisiä funktioita. Prototype yksinkertaistaa JavaScript-ohjelmointia tekemällä sivujen käsittelystä intuitiivisempaa. Prototype helpottaa myös Ajax-kutsujen tekemistä yksinkertaistamalla kutsujen käsittelyä ja huolehtimalla selainyhteensopivuudesta. Prototypen liittäminen WWW-sivulle tapahtuu lisäämällä viittaus JavaScript-tiedostoon sivun otsikkotietoihin. Yleensä Prototype säilyy selaimen välimuistissa eikä sitä tarvitse ladata jokaisella sivulla uudestaan. [17]

Prototype perustuu MIT-lisenssiin. Lisenssin mukaan sitä saa rajattomasti käyttää, kopioida, muokata, yhdistellä, julkaista, levittää, lisensoida ja myydä. Ohjelman kehittäjät eivät kuitenkaan anna minkäänlaisia takuita ohjelmalle. [18]

5.2.2 JavaScript Image Cropper UI

JavaScript Image Cropper UI -kirjasto antaa mahdollisuuden rajata WWW-sivuilla olevista kuvista alueita samaan tapaan kuin kuvankäsittelyohjelmissa. JavaScript Image Cropper UI on rakennettu Prototype-kehyksen ja sen lisäkkeen, script.aculo.us-kirjaston, päälle. Script.aculo.us lisää Prototype-kehykseen dynaamisia visuaalisia efektejä ja vuorovaikutustoimintoja. Kuten Prototypen myös JavaScript Image Cropper UI:n ja script.aculo.us:n liittäminen WWW-sivulle tapahtuu lisäämällä viittaukset JavaScript-tiedostoihin sivun otsikkotietoihin. [19]

JavaScript Image Cropper UI on BSD-lisenssin alainen, joten sitä saa käyttää myös rahallisen hyödyn tavoittelemiseen. Lisenssi vaatii ainoastaan, että kirjastoa käyttävää sovellusta levitettäessä lähdekoodien mukana säilytetään lisenssitekstiä, sovellus näyttää tekijänoikeushuomautuksen ja sovelluksen mainostamisessa ei käytetä kirjaston tekijän nimeä [20]. Script.aculo.us puolestaan on MIT-lisenssin alainen, kuten Prototype [21].

5.2.3 JpGraph

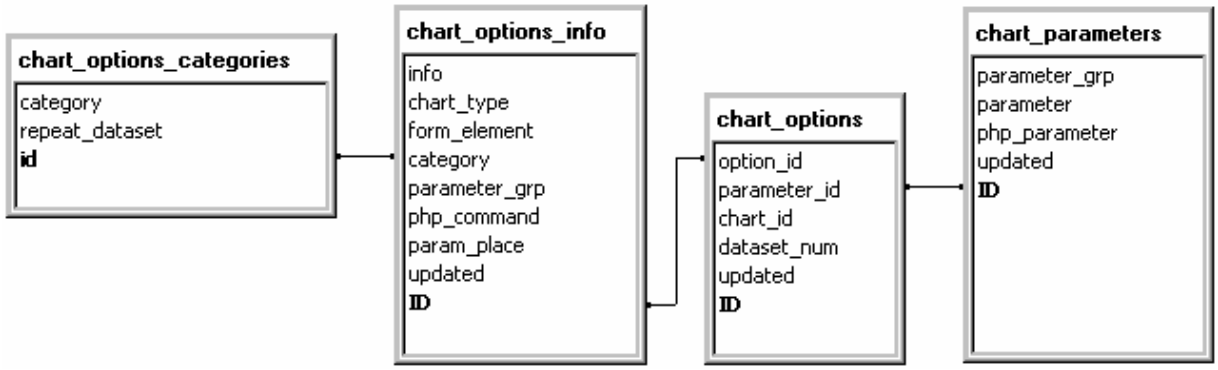
JpGraph on PHP-kielellä kirjoitettu kirjasto, joka luo erilaisia kaaviokuvia. Kaaviokuvien tietojen ja ulkoasun määrittely tapahtuu kirjaston olioiden jäsenfunktioita peräkkäin kutsumalla. Kaaviokuvat voidaan piirtää bittikarttakuviksi. JpGraph-kirjaston liittäminen WWW-sivulle tapahtuu sisällyttämällä tarvittavat tiedostot PHP-koodissa. [22]

JpGraphin käyttäminen mihin tahansa, mistä on rahallista hyötyä, vaatii Pro-lisenssin ostamisen. Rahallinen hyöty ei tarkoita pelkästään sitä, että komponenttia käyttävää sovellusta myydään. Rahallista hyötyä voi saada esimerkiksi siitä, että ohjelmaa käytetään maksullisilla verkkosivuilla tai ohjelmaa käytetään tehostamaan jonkin yrityksen liiketoimintaa. Tavallisella Pro-lisenssillä JpGraphia saa käyttää vain yhdellä palvelimella, ja kalliimmalla Bulk-lisenssillä kirjastoa saa sekä käyttää myytävien tuotteiden osana että käyttää useilla palvelimilla. Pro-lisenssin hinta on 85 euroa ja Bulk-lisenssin 650 euroa. [23]

5.3 Tietokannan taulut

Tietokantaan kuuluu tilastotiedot sisältävä taulu sekä tyylimuotoilut sisältäviä tauluja. Tyylimuotoilut sisältävien taulujen avulla voidaan ohjelmoimalla muotoilla kaaviokuvia. Tämän lisäksi tauluista oli luotava rakenteeltaan sellaisia, että niistä voidaan muodostaa muotoilujen määrittelyn mahdollistava sivu.

Tyylimuotoilut sisältäviä tauluja on neljä: kategoriat sisältävä taulu (chart_options_categories), asetusten määrittelyt sisältävä taulu (chart_options_info), vaihtoehdot sisältävä taulu (chart_parameters) ja muotoiluasetukset sisältävä taulu (chart_options). Taulujen rakenteet ja relaatiot on nähtävissä kuvasta 2. Taulujen luominen ja esimerkkiasetusten syöttäminen tietokantaan SQL-lausein on kirjattu liitteeseen 1. Taulujen kentät ja tehtävät esitellään tarkemmin tässä luvussa.



Kuva 2. Tyylimuotoilut sisältävien tietokannan taulujen rakenteet ja relaatiot.

5.3.1 Tilastotiedot sisältävä taulu

Tilastotiedot sisältävä taulu sisältää pisteitä, joiden mukaan kaaviokuvia voidaan piirtää. Tilastotiedot sisältävää taulua ei esitellä tarkasti, sillä sen rakenne ei ole sovelluksen kannalta oleellinen. Välttämättömän tärkeää taulussa ovat ainoastaan pisteen koordinaatit. Pisteen y-koordinaatti voi olla esimerkiksi jonkin mittauksen arvo. Pisteen x-koordinaatti voi olla esimerkiksi aika, jolloin mittaus on tehty. Vaihtoehtoisia kenttiä taulussa voivat olla mittauksen lähteen ja tyypin tiedot, joiden avulla tilastotietoja voidaan rajata tehokkaammin.

5.3.2 Kategoriat sisältävä taulu

Kategoriat sisältävässä taulussa määritellään valikoiden kategorioita. Kategorioilla on nimi ja tunniste sekä kenttä, joka kertoo liittyykö kategoria tietojoukkoihin vai koko kaaviokuvaan. Ne kategoriat, jotka liittyvät tietojoukkoihin, täytyy toistaa, sillä jokaisella tietojoukolla voi olla eri asetukset. Taulun rakenne on nähtävissä alla.

Kentän nimi	**Kentän selite**
Category	Kategorian nimi.
Repeat_dataset	Kategorian toisto.
Id	Kategorian tunniste.

Taulukko 1. Kategoriat sisältävän taulun rakenne.

5.3.3 Asetusten määrittelyt sisältävä taulu

Asetusten määrittelyt sisältävässä taulussa määritellään asetukset, joilla kaaviokuvia voidaan muotoilla. Asetukset kuuluvat johonkin kategoriataulun kategorioista, joiden mukaan asetukset voidaan ryhmitellä. Oleellista taulussa on, että sen avulla voidaan sekä luoda valikoita että ohjelmoida kaaviokuvan piirtoa. Kaaviokuvien piirtoa varten taulussa on kenttä, jossa on asetuksen PHP-komento. Taulun rakenne on nähtävissä alla.

Kentän nimi	Kentän selite
Info	Asetuksen sanallinen kuvaus.
Chart_type	Kaaviokuvan tyyppi.
Form_element	HTML-elementti, jolla asetusta säädetään.
Category	Kategorian tunniste.
Parameter_grp	Asetuksen vaihtoehtojen ryhmän tunniste.
Php_command	PHP-komento, joka vastaa asetusta.
Param_place	Komentoon liittyvän sijainnin tunniste.
Updated	Päivitysaika.
Id	Asetuksen tunniste.

Taulukko 2. Asetusten määrittelyt sisältävän taulun rakenne.

5.3.4 Vaihtoehdot sisältävä taulu

Vaihtoehdot sisältävässä taulussa määritellään valittavissa olevat vaihtoehdot asetuksille. Toisilleen vaihtoehtoiset vaihtoehdot kuuluvat samaan ryhmään. Taulussa on myös kenttä, joka määrittelee PHP-komennon parametrit, jotka vastaavat vaihtoehtoa. Taulun rakenne on nähtävissä alla.

Kentän nimi	Kentän selite
Parameter_grp	Vaihtoehtojen ryhmän tunniste.
Parameter	Vaihtoehdon sanallinen kuvaus.
Php_parameter	PHP-komennon parametrit, jotka vastaavat vaihtoehtoa.
Updated	Päivitysaika.
Id	Vaihtoehdon tunniste.

Taulukko 3. Vaihtoehdot sisältävän taulun rakenne.

5.3.5 Muotoiluasetukset sisältävä taulu

Muotoiluasetukset sisältävä taulu sisältää käyttäjän asettamia muotoiluja. Taulun avulla voidaan luoda joukkoja muotoiluja, joiden avulla kaaviokuvia voidaan muotoilla. Käytännössä taulussa yhdistetään asetuksia vaihtoehtoihin. Asetusten ja vaihtoehtojen parit liittyvät aina tiettyyn muotoilujoukkoon ja tietojoukkoon. Taulun rakenne on nähtävissä alla.

Kentän nimi	**Kentän selite**
Option_id	Asetuksen tunniste.
Parameter_id	Vaihtoehdon tunniste.
Chart_id	Muotoilujoukon tunniste.
Dataset_num	Tietojoukon järjestysluku.
Updated	Päivitysaika.
Id	Muotoiluasetuksen tunniste.

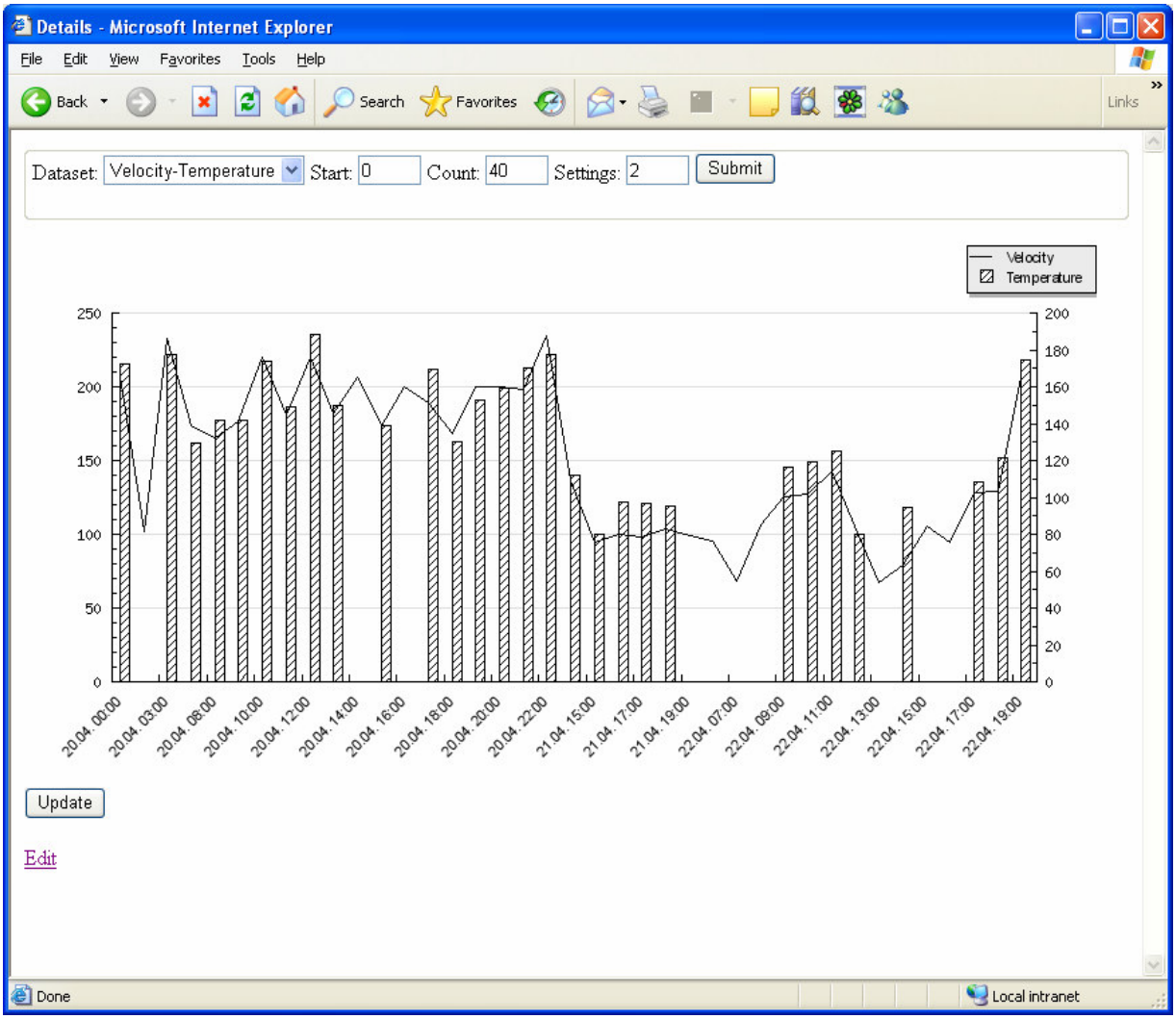
Taulukko 4. Muotoiluasetukset sisältävän taulun rakenne.

5.4 Kaaviokuvia esittävä sivu

Kaaviokuvia esittävällä sivulla on lomake, jolla voidaan rajata tietoja. Sovelluksen tilasta riippuen sivulla voi olla myös kaaviokuva tiedoista. Ennen tietojen rajaamista kaaviokuva ei ole näkyvissä. Esimerkki kaaviokuvia esittävästä sivusta on nähtävissä kuvassa 3. Kuvassa käyttäjä on rajannut tiedot ylimpänä olevalla lomakkeella ja saanut näkyviin kaaviokuvan. Käyttäjä on valinnut kaaviokuvan tyyliasetuksiksi tunnuksen kaksi määrittelemät asetukset, joiden mukaan ensimmäinen tietojoukko esitetään viivadiagrammina ja toinen pylväsdiagrammina.

Lomakkeeseen syötettyjen arvojen perusteella voidaan määritellä tilastotiedot rajaava SQL-kysely, joka tallennetaan sellaisenaan tietokantaan myöhempää käyttöä varten. Tyyliasetusten tunnusta ei kuitenkaan sisällytetä SQL-kyselyyn, vaan se tallennetaan evästeeseen. Lomake on

siten täysin erillään kaaviokuvien piirrosta. Lomakkeen lähettäminen palauttaa käyttäjän samalle sivulle, mutta SQL-kyselyn tunniste annetaan sivulle parametrina.



Kuva 3. Esimerkki kaaviokuvia esittävästä sivusta.

Lomakkeen alapuolelle ilmestyy kaaviokuva ja siihen liittyvät painikkeet, jos SQL-kyselyn tunniste on annettu sivulle parametrina. Kaaviokuvan piirto tapahtuu erillään sivusta, sillä kuva on erillinen tiedosto. Kaaviokuvan piirron suorittava komentosarja tarvitsee parametreina tilastotiedot rajaavan SQL-kyselyn tunnisteen ja muotoiluasetuskomentojen tunnisteen.

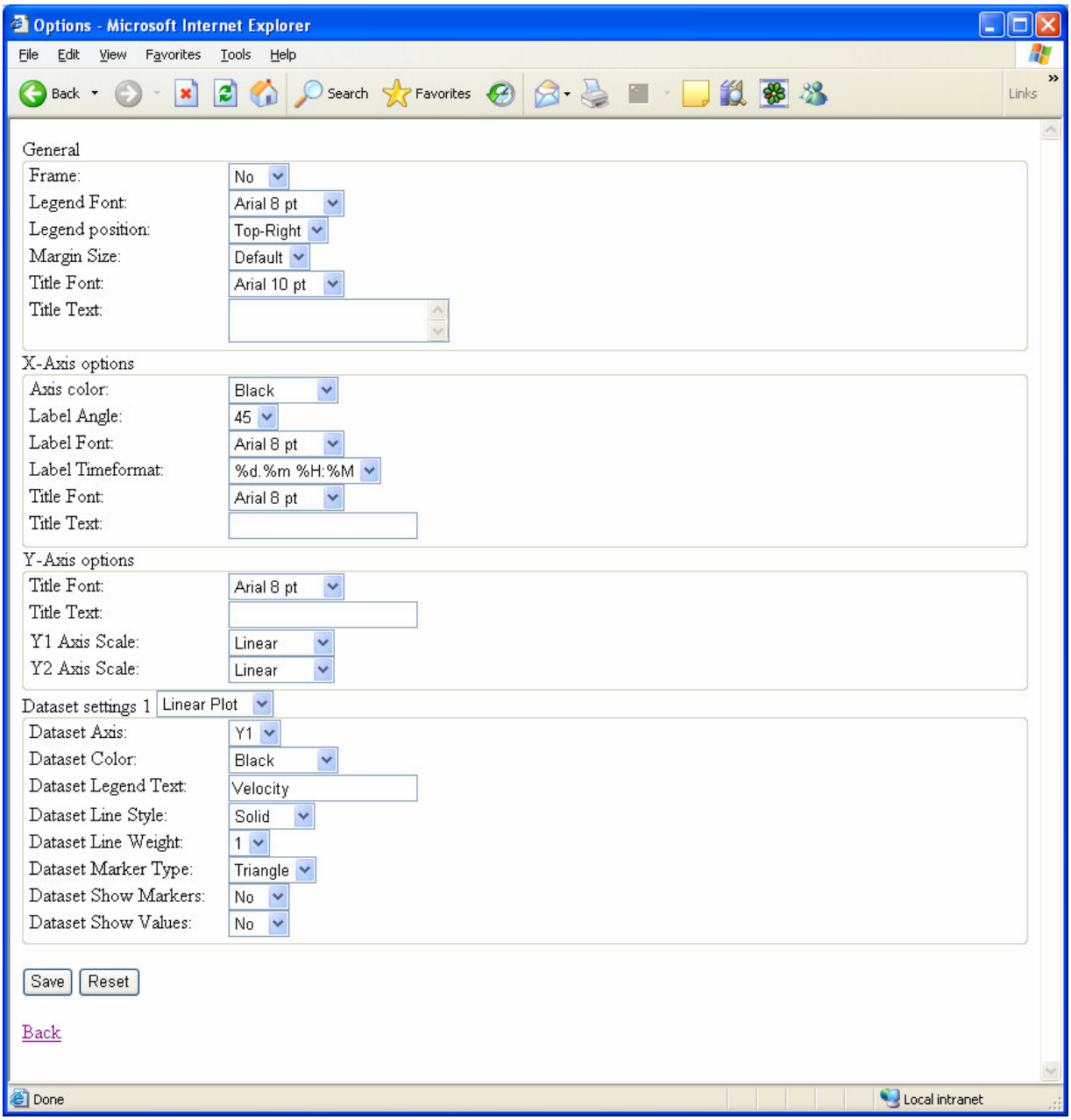
Kaaviokuvan piirrossa suoritetaan ensin SQL-kysely, joka saadaan tunnisteen avulla. SQL-kyselyllä saadut tilastotiedot asetetaan taulukoihin. Tämän jälkeen luodaan kaaviokuvaolio. Kaaviokuvaolion tyyliä muokataan tietokannassa olevien muotoiluasetuskomentojen perusteella. Kaaviokuvaan lisätään kaikki tietojoukot, jotka on määritelty SQL-kyselyssä. Myös tietojoukkoja muokataan tietokannassa olevien muotoiluasetuskomentojen perusteella. Lopulta kaaviokuva piirretään bittikarttakuvaksi ja lähetetään selaimelle.

Kaaviokuvaa voidaan rajata hiiren avulla. Rajaamisen jälkeen on painettava päivityspainiketta, jolloin sivulla oleva kuva asetetaan JavaScript-komentosarjan avulla uudestaan ladattavaksi. JavaScript Image Cropper UI:n avulla hiirellä valitut kuvan koordinaatit saadaan selville, ja ne lähetetään kaaviokuvan piirtävälle komentosarjalle, joka päättelee koordinaattien perusteella todelliset rajat. Sivulla oleva kuva vaihtuu ilman koko sivun uudelleen lataamista.

5.5 Muotoilujen määrittelyn mahdollistava sivu

Muotoilujen määrittelyn mahdollistavalla sivulla on lomake, joka tuotetaan aina tietokannan määrittelyjen mukaan. Lomakkeella näkyy käyttäjän evästeessä sijaitsevan tunnuksen perusteella määritellyt muotoiluasetukset. Esimerkki muotoilujen määrittelyn mahdollistavasta sivusta on nähtävissä kuvassa 4.

Lomakkeen valikot järjestellään aina kategorioiden mukaan. Kategoriat luodaan erillisessä komentosarjatiedostossa, sillä kategorioiden täytyy muuttua sivun käytön aikana tietojoukon diagrammin tyyppiä muutettaessa. Diagrammien tyypeillä on erilaisia muotoiluvaihtoehtoja, joten tiettyyn diagrammityyppiin kuulumattomia vaihtoehtoja ei kannata näyttää käyttäjälle. Kategorioiden sisältämät valikot vaihtuvat sivua lataamatta Ajax-funktion avulla.



Kuva 4. Esimerkki muotoilujen määrittelyn mahdollistavasta sivusta.

Lomaketta tallennettaessa tiedot lähetetään erilliselle komentosarjalle. Komentosarja huolehtii asetusten tallentamisesta tietokantaan. Tallentamisen jälkeen ilmoitus näytetään käyttäjälle, joka voi jatkaa muotoilujen määrittelyä tai palata kaaviokuvia piirtävälle sivulle.

6 JOHTOPÄÄTÖKSET

Työssä esiteltiin tilastotiedoista kaaviokuvia piirtävä WWW-sovellus. Kaaviokuvat esitetään WWW-sivulla, jolla myös kaaviokuvien tiedot voidaan rajata. Sovelluksen esittämien kaaviokuvien muotoiluja voidaan hallita erillisen muokkaussivun avulla. Hallittavissa olevien muotoilujen sekä niiden vaihtoehtojen määrä riippuu sovelluksen tietokannan laajuudesta. Sovellus kehitettiin PSPad-tekstinkäsittelyohjelmaa käyttäen. Ohjelmointikielinä käytettiin PHP:ta ja JavaScriptiä. Sovelluksen kaaviokuvia piirtävänä komponenttina toimii lisensoitu JpGraph-kirjasto. JavaScript-ohjelmoinnissa käytettiin apuna avoimen lähdekoodin Prototype-kehystä ja JavaScript Image Cropper UI -kirjastoa. Muut osat sovelluksesta ohjelmoitiin itse.

Ohjelmointikielten valintaan voi vaikuttaa hyvinkin paljon se, mitä vaatimuksia kielillä on alustoilta. Palvelinpuolen kielet vaativat aina tulkin tai vastaavan ohjelmiston asentamisen palvelimelle. Jos palvelimelle on jo asennettu jokin tulkki, sen käyttäminen uuden sovelluksen kielenä on helpointa. Palvelinpuolen kielen vaihtaminen on kuitenkin sitä perustellumpaa, mitä merkittävämmästä kehitysprojektista on kyse. Tässä työssä väliohjelmiston tekniikaksi valittiin PHP, koska se oli palvelimella jo ennestään käytössä. Palvelinpuolen kielen valitsemisen seuraukset tiedetään aina selvästi, mutta asiakaspuolen tapauksessa seurauksia ei välttämättä tiedetä. Jos sovellus vaatii asiakaspuolelta tukea jollekin tekniikalle, rajattu käyttäjäjoukko saattaa pystyä käyttämään sovellusta, mutta osa käyttäjistä välttämättä ei. Vaikka asiakaspuolen tekniikoiden tuen asentaminen on ilmaista, se ei välttämättä onnistu. Monet käyttäjät eivät osaa asentaa tuen tarjoavia lisäkeohjelmia selaimeen ja joillakin työpaikoilla lisäkeohjelmien asentaminen tietokoneisiin on jopa estetty. Yhteensopivuus asiakaskoneiden kanssa saadaan taattua varmimmin käyttämällä kevyen asiakkaan arkkitehtuuria, jolloin asiakaskoneilta vaaditaan vain selain. Tällöin monet käytettävyyttä ja monipuolisuutta edistävät ominaisuudet eivät ole ohjelmoijan käytettävissä. Työssä tehdyn sovelluksen rajoitukset asiakaspuolen tekniikoille olivat tiedossa, sillä sovelluksen

käyttäjäkunta oli rajallinen. Raskaimmat asiakaspuolen tekniikat, Java ja Flash, eivät olleet käytössä, mutta JavaScript oli.

Ohjelmointikielten ja tekniikoiden tarjoamat ominaisuudet saattavat vaikuttaa sovelluksen tekniikoiden valintaan. Esimerkiksi Javan ajatellaan olevan hyvä palvelinpuolen tekniikka sen alustariippumattomuuden takia. Javalla kirjoitettuja ohjelmia voidaan ajaa lähes millä tahansa alustalla. Jotkut sovellukset vaativat jo luonteensa puolesta nimenomaan tätä ominaisuutta. Tässä työssä tekniikoiden valintaan vaikutti se, mille tekniikoille valmiita kaaviokuvia piirtäviä kirjastoja oli olemassa. PHP:n lisäksi JpGraphia vastaavia kirjastoja on olemassa muun muassa Javalle ja Flashille. Ohjelmointikielten valintaan vaikuttaa usein myös se, mitä taitoja sovelluksen kehittäjillä on. Yleensä kehittäjät valitsevat jonkin osaamistaan kielistä, mikäli heillä on tilaisuus valita. Uuden kielen opettelu on usein epämukavaa ja aikaa vievää. Microsoftin .Net-kehyksen eräs merkittävimmistä vahvuuksista on se, että se mahdollistaa useiden ohjelmointikielten käytön. Kehittäjä voi valita ohjelmointikielistä sen, jonka osaa parhaiten, vaikka osa sovelluksesta olisi kehitetty muilla kielillä.

Työn laajuus vaikutti siihen, mitä apumenetelmiä kehityksessä kannatti käyttää. Oli vaikeaa kuvitella mallinnuskielten käyttämisestä saatavan hyötyä suunnittelussa. Mallinnuskielten hyötyjä ei tosin tutkittu työn aikana. Myös monimutkaisuus vaikuttaa uusien työkalujen käyttöönottoon. Liian monimutkaisia työkaluja ei kannata opetella ennen kuin siitä saatava hyöty on suurempi kuin opetteluun kuluva aika. Kokematon käyttäjä joutuu turvautumaan yksinkertaisiin työkaluihin, joilla päästään nopeasti edistämään varsinaista työtä. Työssä käytettiin onnistuneesti yksinkertaista tekstinkäsittelyohjelmaa, joka korosti lähdekoodien tekstejä. Tekstin korostamisesta on kehittämisessä selvästi apua, mutta sitä ei mitattu.

Valmiiden komponenttien hyödyntäminen aikaisti reilusti sovelluksen valmistumista. Työssä tehdyn sovelluksen noin kolmesta megatavusta lähdekoodia vain noin 50 kilotavua täytyi tehdä itse. Lähdekoodien kirjoittamiseen kului hieman yli kolme 40-tuntista työviikkoa. Vain osaa valmiiden komponenttien lähdekoodeista käytetään sovelluksessa, mutta siitä huolimatta

valmiiden komponenttien tuoma hyöty oli suuri. JpGraph-kirjastoa lukuun ottamatta valmiit komponentit olivat avointa lähdekoodia. Lisenssien tulkitsemisen vaikeuden sanotaan olevan eräs avoimen lähdekoodin kiusallisimmista ongelmista. Avoimen lähdekoodin käyttäjän on tutkittava hyvin tarkasti mihin tarkoituksiin komponentteja saa käyttää – varsinkin jos komponentteja käytetään rahallisen hyödyn tavoittelemiseen. Työssä käytettyjen avoimen lähdekoodin ohjelmien tekijät kertoivat WWW-sivustoillaan selkokielisesti, mihin heidän ohjelmiaan saa käyttää.

Eräs työn tavoitteista oli tehdä sovelluksesta helposti laajennettava, sillä kaikkien mahdollisten ominaisuuksien lisääminen kerralla olisi ollut liian monimutkaista suunnittelun kannalta. Sovelluksella haluttiin voitavan muokata kaaviokuvien muotoiluja, mutta käyttäjien tarvitsemat vaihtoehdot tulevat esille yksitellen pitkän ajan kuluessa. Sovelluksen laajentamisen haluttiin olevan tehtävissä Don’t Repeat Yourself -periaatteen mukaisesti yhdestä keskitetystä paikasta, jolloin sovelluksen osien välisistä riippuvuuksista ei tarvitsisi huolehtia sitä laajennettaessa. Ongelmana oli, että laajennukset vaikuttavat sekä kaaviokuvien muokkauksen mahdollistavaan sivuun että kaaviokuvan piirrossa käytettäviin PHP-komentoihin. Ongelma ratkaistiin tekemällä sovelluksesta tietokannalla ohjelmoitava. Muotoiluvaihtoehtojen lisääminen sovellukseen onnistuu tietokannan tauluihin rivejä lisäämällä. Käyttäjien tietokantaan tallentamat yksittäisten kaaviokuvien muotoiluasetusten kokonaisuudet eivät tule käyttökelvottomiksi sovellusta laajennettaessa, jos laajentaminen tehdään oikein. Sekä kaaviokuvien muokkauksen mahdollistava sivu että kaaviokuvia piirtävä sivu luodaan jokaisella sivulatauksella samojen tietokannan taulujen perusteella. Sivujen rakenteista voitiin siten tehdä mahdollisimman lyhyitä ja yleiskäyttöisiä.

Sovellusta voisi kehittää jatkossa erilaisin tavoin. Sovellusta voisi kehittää tekemällä sovelluksen laajentamiseen sopiva työkalu. Rivien lisääminen tietokannan tauluihin on altista virheille, mikä voisi olla estettävissä hyvin suunnitellun työkalun avulla. Työkalu myös nopeuttaisi sovelluksen laajentamista. Työkalu voisi auttaa ainakin varmistamalla, että JpGraph-kirjaston olioiden jäsenfunktioiden kutsut ja parametrit on kirjoitettu oikein.

Sovelluksen käytettävyyttä voitaisiin myös parantaa. Työn puitteissa ei ollut mahdollista tarkkailla käyttäjiä käyttämässä sovellusta. Käytettävyystestit olisivat tuoneet esille asioita, joita kehittäjä ei itse havainnut. Käytettävyyttä voisi mahdollisesti parantaa tekemällä käyttäjäryhmiä. Eri käyttäjäryhmille tarjottaisiin eri määrä muotoiluasetuksia muokattavaksi.

LÄHTEET

1. Feinstein, Wei. A Study of Technologies for Client/Server Applications. Teoksessa: Geist, Robert (toim.). ACM Southeast Regional Conference: Proceedings of the 38th annual on Southeast regional conference. Clemson, South Carolina: ACM Press, 2000. S. 184–193. ISBN 1-58113-250-6.

2. Chaudbury, Abhijit & Rao, H. Raghav. Introducing Client/Server Technologies in Information Systems Curricula. ACM SIGMIS Database, 1997. Vol. 28: 4. S. 20–32. ISSN 0095-0033.

3. Conallen, Jim. Modeling Web Application Architectures with UML. Communications of the ACM, 1999. Vol. 42: 10. S. 63–70. ISSN 0001-0782.

4. Makarenya, Diana. Web Software Engineering – the value added approach. Diplomityö. Lappeenrannan teknillinen yliopisto, Tietotekniikan osasto. Lappeenranta, 2003. 82 s.

5. Murugesan, San et al. Web Engineering: A New Discipline for Development of Web-based Systems. Teoksessa: 5, San & Deshpande Yogesh (toim.). Lecture Notes in Computer Science; Vol. 2016: Web Engineering, Software Engineering and Web Application Development. London, UK: Springer Verlag, 2001. S. 3–13. ISBN 3-540-42130-0.

6. Veen, Jeffrey. Inside Web Design. Kääntänyt Timo Haanpää. Jyväskylä: Edita Publishing, 2002. 258 s. ISBN 951-826-492-9.

7. Li, Jingfeng; Chen, Jian & Chen, Ping. Modeling Web Application Architecture with UML. Teoksessa: Titsworth, Frances (toim.). Proceedings of the 36th International

Conference on Technology of Object-Oriented Languages and Systems (TOOLS ’00). Xi’an, Kiina: IEEE Computer Society, 2000. S. 265–274. ISBN 0-7695-0875-8.

8. Keränen, Vesa; Lamberg, Niko & Penttinen Jukka. Web-julkaiseminen & multimedia. Jyväskylä: Docendo Finland (julk.) & SanomaWSOY (kust.), 2006. 272 s. ISBN 951-846-284-4.

9. Preciado, J et al. Necessity of methodologies to model Rich Internet Applications. Teoksessa: Distanto, Damiane (toim.). Proceedings of the 2005 Seventh IEEE International Symposium on Web Site Evolution (WSE’05). Los Alamitos, California: IEEE Computer Society, 2005. S. 7–13. ISBN 0-7695-2470-2.

10. Yank, Kevin. Which Server Side Language Is Right for You [verkkodokumentti]. 2001 [viitattu 24.5.2007]. 7 WWW-sivua. Saatavissa: http://www.sitepoint.com/article/server-side-language-right/ (etusivu).

11. Paulson, Linda. Building Rich Web Applications with Ajax. Teoksessa: Computer, 2005. Vol. 38: 10. S. 13–17. ISSN 0018-9162.

12. Ossher, Harold; Harrison, William & Tarr, Peri. Software Engineering Tools and Environments: a Roadmap. Teoksessa: Fin, Anthony (toim.). Proceedings of the Conference on the Future of Software Engineering. Limerick, Ireland: ACM Press, 2000. S. 261–277. ISBN 1-58113-253-0.

13. Kilpeläinen, Terhi. Avoin lähdekoodi ohjelmistotalojen näkökulmasta. Diplomityö. Lappeenrannan teknillinen yliopisto, Tietotekniikan osasto. Lappeenranta, 2005. 69 s.

14. Tiemann, Michael. Licenses by Name [verkkodokumentti]. Open Source Initiative, 2006 [viitattu 10.5.2007]. Saatavissa: http://opensource.org/licenses/alphabetical.

15. Seppänen, Marko. Thoughts on Competitive Strategy and OS. Teoksessa: Helander, Nina & Mäntymäki, Maria (toim.). Empirical Insights on Open Source Business. Tampere, Tampere University of Technology (TUT) and University of Tampere (UTA), 2006. S. 4–10. EBRC Research Reports 34. ISBN 952-15-1612-7 (TUT) 951-44-6669-1 (UTA).

16. Kuoppala, Hannu et al. Käytettävyyden psykologia. Helsinki: IT Press (julk.) & Edita (kust.), 2002. 343 s. ISBN 951-826-574-7.

17. Stephenson, Sam. Prototype Tips and Tutorials [verkkodokumentti]. 2006 [viitattu 23.5.2007]. 4 WWW-sivua. Saatavissa: http://www.prototypejs.org/learn (etusivu).

18. Stephenson, Sam. Prototype JavaScript Framework: License [verkkodokumentti]. 2005 [viitattu 10.5.2007]. Saatavissa: http://www.prototypejs.org/license.

19. Spurr, Dave. JavaScript Image Cropper UI, using Prototype & script.aculo.us [verkkodokumentti]. 2006 [viitattu 23.5.2007]. Saatavissa: http://www.defusion.org.uk/code/javascript-image-cropper-ui-using-prototype-scriptaculous/.

20. Open Source Initiative OSI. The BSD License [verkkodokumentti]. 2006 [viitattu 23.5.2007]. Saatavissa: http://www.opensource.org/licenses/bsd-license.php.

21. Fuchs, Thomas. License [verkkodokumentti]. 2005 [viitattu 10.5.2007]. Saatavissa: http://wiki.script.aculo.us/scriptaculous/show/License.

22. Aditus Consulting. JpGraph – PHP Graph Creating Library [verkkodokumentti]. Julkaisuaika tuntematon [viitattu 23.5.2007]. Saatavissa: http://www.aditus.nu/jpgraph/.

23. Aditus Consulting. JpGraph Professional Version [verkkodokumentti]. Julkaisuaika tuntematon [viitattu 10.5.2007]. Saatavissa: http://www.aditus.nu/jpgraph/proversion.php.

LIITE 1. TIETOKANNAN TAULUJEN LUOMINEN JA LÄHTÖASETUKSET

```
CREATE TABLE `chart_options_categories` (
  `category` varchar(255) NOT NULL,
  `repeat_dataset` tinyint(1) NOT NULL,
  `id` tinyint(3) unsigned NOT NULL auto_increment,
  PRIMARY KEY  (`id`)
);

INSERT INTO `chart_options_categories` (`category`, `repeat_dataset`, `id`)
VALUES
('General', 0, 1),
('X-Axis options', 0, 2),
('Y-Axis options', 0, 3),
('Dataset settings', 1, 11);


CREATE TABLE `chart_options_info` (
  `info` varchar(100) NOT NULL,
  `chart_type` set('line','bar','scatter','pie') NOT NULL,
  `form_element` set('text','select','checkbox','textarea') NOT NULL,
  `category` tinyint(3) NOT NULL,
  `parameter_grp` tinyint(4) NOT NULL,
  `php_command` varchar(256) NOT NULL,
  `param_place` tinyint(3) NOT NULL,
  `updated` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `ID` tinyint(3) unsigned NOT NULL auto_increment,
  PRIMARY KEY  (`ID`)
);

INSERT INTO `chart_options_info` (`info`, `chart_type`, `form_element`,
`category`, `parameter_grp`, `php_command`, `param_place`, `updated`, `ID`)
VALUES
('Dataset Color', 'line,bar', 'select', 11, 3, 'SetColor', 3, '0000-00-00
00:00:00', 1),
('Dataset Axis', 'line,bar,scatter', 'select', 11, 7, '', 4, '0000-00-00
00:00:00', 6),
('Dataset Type', 'line,bar,scatter', 'select', 11, 10, '', 2, '0000-00-00
00:00:00', 10),
('Title Font', 'line,bar,scatter', 'select', 1, 2, 'title->SetFont', 1,
'0000-00-00 00:00:00', 12),
('Label Font', 'line,bar,scatter', 'select', 2, 2, 'xaxis->SetFont', 1,
'0000-00-00 00:00:00', 14),
('Y1 Axis Scale', 'line,bar,scatter', 'select', 3, 11, 'SetScale', 5,
'0000-00-00 00:00:00', 16),
('Frame', 'line,bar,scatter', 'select', 1, 8, 'SetFrame', 1, '0000-00-00
00:00:00', 17),
```

(jatkuu)

(liite 1 jatkoa)

```
('Margin Size', 'line,bar,scatter', 'select', 1, 12, 'SetMargin', 1, '0000-
00-00 00:00:00', 18),
('Legend position', 'line,bar,scatter', 'select', 1, 13, 'legend->Pos', 1,
'0000-00-00 00:00:00', 19),
('Title Font', 'line,bar,scatter', 'select', 3, 2, 'yaxis->title->SetFont',
1, '0000-00-00 00:00:00', 21),
('Dataset Bar Fill Color', 'bar', 'select', 11, 3, 'SetFillColor', 3,
'0000-00-00 00:00:00', 22),
('Label Angle', 'line,bar,scatter', 'select', 2, 15, 'xaxis-
>SetLabelAngle', 1, '0000-00-00 00:00:00', 24),
('Label Timeformat', 'line,bar,scatter', 'select', 2, 16, 'xaxis-
>SetTickLabels', 1, '0000-00-00 00:00:00', 27);


CREATE TABLE `chart_parameters` (
  `parameter_grp` tinyint(4) NOT NULL,
  `parameter` varchar(100) NOT NULL,
  `php_parameter` varchar(255) NOT NULL,
  `updated` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `ID` int(10) unsigned NOT NULL auto_increment,
  PRIMARY KEY  (`ID`)
);

INSERT INTO `chart_parameters` (`parameter_grp`, `parameter`,
`php_parameter`, `updated`, `ID`) VALUES
(2, 'Arial 8 pt', 'FF_ARIAL,FS_NORMAL,8', '0000-00-00 00:00:00', 1),
(2, 'Arial 10 pt', 'FF_ARIAL', '0000-00-00 00:00:00', 2),
(3, 'Black', '"black"', '0000-00-00 00:00:00', 5),
(3, 'Royal Blue', '"royalblue"', '0000-00-00 00:00:00', 10),
(3, 'Brown', '"brown"', '0000-00-00 00:00:00', 11),
(3, 'Sea Green', '"seagreen"', '0000-00-00 00:00:00', 12),
(7, 'Y1', 'Add', '0000-00-00 00:00:00', 29),
(7, 'Y2', 'AddY2', '0000-00-00 00:00:00', 30),
(8, 'No', 'false', '0000-00-00 00:00:00', 31),
(8, 'Yes', 'true', '0000-00-00 00:00:00', 32),
(10, 'Linear Plot', 'new LinePlot', '0000-00-00 00:00:00', 36),
(10, 'Bar Plot', 'new BarPlot', '0000-00-00 00:00:00', 37),
(10, 'Scatter Plot', 'new ScatterPlot', '0000-00-00 00:00:00', 38),
(11, 'Linear', '"textlin"', '0000-00-00 00:00:00', 39),
(1, '', '""', '0000-00-00 00:00:00', 40),
(12, 'Default', '65,45,50,75', '0000-00-00 00:00:00', 41),
(13, 'Top-Right', '0.00, 0.00, "right", "top"', '0000-00-00 00:00:00', 42),
(11, 'Logaritmic', '"textlog"', '0000-00-00 00:00:00', 44),
(15, '45', '45', '0000-00-00 00:00:00', 45),
```

(jatkuu)

(liite 1 jatkoa)

```
(15, '90', '90', '0000-00-00 00:00:00', 46),
(16, '%d.%m %H:%M', 'convert_times("%d.%m. %H:%M")', '0000-00-00 00:00:00',
50),
(16, '%H:%M', 'convert_times("%H:%M")', '0000-00-00 00:00:00', 51);


CREATE TABLE `chart_options` (
  `option_id` tinyint(3) NOT NULL,
  `parameter_id` tinyint(3) NOT NULL,
  `chart_id` tinyint(4) NOT NULL,
  `dataset_num` tinyint(3) NOT NULL,
  `updated` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `ID` int(10) unsigned NOT NULL auto_increment,
  PRIMARY KEY  (`ID`)
);

INSERT INTO `chart_options` (`option_id`, `parameter_id`, `chart_id`,
`dataset_num`, `updated`) VALUES
(12, 2, 1, 0, '0000-00-00 00:00:00'),
(14, 1, 1, 0, '0000-00-00 00:00:00'),
(16, 39, 1, 0, '0000-00-00 00:00:00'),
(17, 31, 1, 0, '0000-00-00 00:00:00'),
(18, 41, 1, 0, '0000-00-00 00:00:00'),
(19, 42, 1, 0, '0000-00-00 00:00:00'),
(24, 45, 1, 0, '0000-00-00 00:00:00'),
(27, 50, 1, 0, '0000-00-00 00:00:00'),
(1, 10, 1, 1, '0000-00-00 00:00:00'),
(6, 29, 1, 1, '0000-00-00 00:00:00'),
(10, 36, 1, 1, '0000-00-00 00:00:00'),
(22, 10, 1, 1, '0000-00-00 00:00:00'),
(1, 11, 1, 2, '0000-00-00 00:00:00'),
(6, 29, 1, 2, '0000-00-00 00:00:00'),
(10, 36, 1, 2, '0000-00-00 00:00:00'),
(22, 11, 1, 2, '0000-00-00 00:00:00'),
(1, 12, 1, 3, '0000-00-00 00:00:00'),
(6, 29, 1, 3, '0000-00-00 00:00:00'),
(10, 36, 1, 3, '0000-00-00 00:00:00'),
(22, 12, 1, 3, '0000-00-00 00:00:00');
```