TEKSTIVIESTIPALVELUN INTEGROINTI SÄHKÖISEEN OPPIMISYMPÄRISTÖÖN

Case:Moodle

LAHDEN AMMATTIKORKEAKOULU
Tekniikan ala
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka
Opinnäytetyö
Kevät 2011
Niko Hast

Lahden ammattikorkeakoulu
Ohjelmistotekniikka

HAST NIKO

Tekstiviestipalvelun integrointi sähköiseen oppimisympäristöön
Case: Moodle

Ohjelmistotekniikan opinnäytetyö, 36 sivua, 5 liitesivua

Kevät 2011

TIIVISTELMÄ

Tämä opinnäytetyö käsittelee Lahden ammattikorkeakoulun Moodle-oppimisalustaan tehtyä tekstiviestipalvelua ja sen ohjelmointia.

Teoriaosuus käsittelee www-ohjelmointiin tarvittavia ohjelmointikieliä, Moodlea yleisesti, Moodlen asentamista sekä käyttöönottoa. Tässä osuudessa käydään läpi myös tekstiviestiteknologia ja sen perusperiaatteet. Tekstiviestipalvelun kehitys on myös tarkastelun alla. Moodlesta käydään läpi myös ohjelmointikäytäntöjä, käyttäjien rooleja sekä itse Moodlen että lisäosien asennus. Teoriaosan lopussa on tarkasteltu tekstiviestien käyttöä yhteisöjen viestinnässä, ja analysointia opiskelijoiden ja opettajien kesken pidetyssä kyselyssä, joka koskee tekstiviestipalvelua Lahden ammattikorkeakoulussa.

Työn tarkoituksena on luoda tekstiviestipalvelu Moodleen, jolla voidaan informoida oppilaita mahdollisista äkkinäisistä kurssimuutoksista tai sairaspoissaoloista sekä yhtenä aspektina kriisiviestintää.

Tekstiviestilisäosan suunnittelu ja toteuttaminen onnistui suunnitelmien mukaan ja käyttäjäkyselyn perusteella kyseinen ominaisuus oli tarpeellinen sekä opettajien että oppilaiden mielestä.

Avainsanat: PHP, Moodle, HTML, XML, SMS

Lahti University of Applied Sciences
Degree Programme in Software Engineering

NIKO HAST

Integrating a text messaging service to a virtual course management system
Case: Moodle

Bachelor's Thesis in Software Engineering 36 pages, 5 appendices

Spring 2011

ABSTRACT

This Bachelor's thesis deals with designing a text messaging service for Lahti University of Applied Sciences for their Moodle course management system and the coding of it.

The goal was to create a text messaging service which can be used to inform students about sudden changes on courses or for example in case of a medical emergency.

The beginning of the theory part discusses the web-programming languages used in the study, Moodle in general and the installing and management of the Moodle installation. Text messaging technology and its principles are also included in the theory part.

In the following part, the different steps of developing the text messaging service for Moodle are described. The study also includes programming practices, user roles and the installation of Moodle and its plugins. At the end of the theory part, text messaging in communities is analyzed and finally there is an analysis of a questionnaire survey made about the text messaging service amongst the students and teachers in the Lahti University of Applied Sciences.

Key words: PHP, Moodle, HMTL, XML, SMS

SISÄLLYS

1 JOHDANTO

Koska sähköpostiviestit ovat tänä päivänä varmasti yleisin tiedonantotapa koulujen sisällä, on tämän lisäksi myös matkapuhelinten yleistyminen tuonut nopean ja tehokkaan tavan informoida opiskelijoita. Tekstiviestien käyttö tuo tiedon perille nopeammin, jolloin reagointi tilanteeseen on helpompaa.

Tämän työn tavoitteena on integroida tekstiviestipalvelu Moodle-oppimisympäristöön siten, että opettajat voivat lähettää tekstiviestejä kurssiensa oppilaille. Tällä hetkellä www-teknologiaan perustuva ilmoitusjärjestelmä, jossa muutoksista kerrotaan www-sivulla, on toimiva, mutta vaatii opiskelijalta paljon aktiivisuutta. Koskaan ei voi tietää, milloin opettaja ilmoittaa mahdollisesta poissaolostaan.

Työssä keskitytään Moodleen ja sen ohjelmointikäytäntöihin, jotta toimivan ja julkaisukelpoisen lisäosan kehittäminen on mahdollista. Lisäosa vaatii myös useita ohjelmointikieliä toimivuuden takaamiseksi, joten www-ohjelmointitekniikoista käydään läpi PHP, HTML, XML ja MySQL. Teoriapuolen lopuksi tarkastellaan tekstiviestien käyttöä nykypäivänä ja muutamia käyttökohteita.

Tässä työssä luvussa kaksi käydään läpi tarvittavat www-teknologiat työn kannalta. PHP:llä kehitellään itse toimivuus lisäosaan, HTML:llä sivun ulkoasu, XML hoitaa lokin kirjoittamisen tiedostoon ja MySQL:llä hoidetaan Moodlen tietokannat sekä lisäosan tietokantaan lokin kirjoittaminen.

Luku kolme keskittyy yleisesti tekstiviestiteknologiaan ja sen toimintaperiaatteisiin. Luku kattaa tekstiviestin lähettämisen periaatteet sekä tekstiviestin sisältöön liittyvät asiat.

Luvussa neljä käydään läpi käyttöympäristön kuvaus. Käyttöympäristöön kuuluvat itse tekstiviestipalvelu, Moodle, sen ohjelmointikäytännöt sekä käyttäjien roolit. Luvussa tarkastellaan myös Moodlen asentamista paikalliselle tietokoneelle testausta varten sekä Moodlen tietokantarakennetta ja lisäosien

asentamista.

Luvussa viisi keskitytään itse lisäosaan ja sen rakenteeseen sekä lisäosan käyttöön Moodlen sisällä. Viimeisenä suoritetaan käyttäjäkysely ja siinä kartoitetaan, miten hyödyllisenä käyttäjät näkevät tekstiviestien käytön.

2 WWW-TEKNOLOGIAT

2.1 PHP

Työssä käytetään PHP-ohjelmointikieltä, versionumeroltaan viisi, joka tulee sanoista PHP: Hypertext Preprocessor. PHP on kehitelty erityisesti dynaamisten web-sivujen luontiin web-palvelinympäristöissä. Kyseessä on komentosarjakieli, jossa itse koodi tulkitaan vasta ohjelman suoritusvaiheessa. PHP on lähes alustariippumaton, ja näin ollen sitä voidaan käyttää useilla alustoilla ja käyttöjärjestelmillä. Rasmus Lerdorf julkaisi ensimmäisen version kehittäjien käyttöön vuonna 1995, ja PHP-kieltä onkin kehitelty siitä lähtien (Appu 2002, 13).

PHP oli alun perin kokoelma WWW-pohjaisten sovelluksien kehittämistä varten tehtyjä helpottavia rutiineja eli tietynlainen palvelimella ajettavien CGI-ohjelmien tekemistä helpottava komentokokoelma. PHP:n komentoja voidaan kirjoittaa suoraan HTML-sivujen sisään. (Heinisuo & Ranta 2007, 12.)

PHP on tulkattava kieli, eli PHP-koodi ajetaan WWW-sivun sisällä joka kerta, kun sivu ladataan palvelimelta. PHP voi olla osana Apache-palvelinta, jolloin se on osa WWW-palvelinta. Tämän vuoksi PHP-koodin ajaminen on suhteellisen nopeaa, eikä ulkoista tulkkia tarvita. Koodi ajetaan aina palvelimella ennen kuin sivu ladataan selaimeen. (Heinisuo & Ranta 2007, 13.)

Selaimessa näkyy ainoastaan ohjelman tulostus, kun sivulla oleva koodi suoritetaan. PHP-koodi on ainoastaan silloin nähtävissä, kun koodin sisältämää tiedostoa käsitellään FTP:n tai tiedostojen jaon kautta. Tästä syystä PHP-koodia sisältävät sivut eivät toimi suoraan tietokoneen levyltä, koska PHP suoritetaan palvelimen päässä ja sivut on ladattava palvelimelta. (Heinisuo & Ranta 2007, 13.)

Työn käytännönosuus koostuu pääosin PHP-ohjelmoinnista, koska Moodle on kehitetty PHP:n avulla. Myös tekstiviestien lähetyksessä käytettävä rajapinta on tehty PHP:lla. Nämä asiat esitellään työn käytännön osuudessa.

2.2 HTML

HTML tulee sanoista Hyper Text Markup Language, eikä se ole ohjelmointikieli, vaan merkintäkieli. Merkintäkieli on joukko tageja, jotka kuvaavat tässä tapauksessa Internet-sivuja. (W3Schools, 2011.)

Merkintätageja kutsutaan HTML-tageiksi ja ne on ympäröity kulmasulkeilla, kuten esimerkiksi <head>. Tageilla on aina alkava tagi ja sulkeva tagi, esimerkiksi <head> </head>. Näitä voidaan kutsua myös avaavana tagina ja sulkevana tagina. Kuvio 1 selventää rakennetta. (W3Schools, 2011.)

```
1  <html>
2  <body>
3      <h1>Ensimmäinen otsake</h1>
4      <p>Ensimmäinen lause.</p>
5  </body>
6  </html>
7
```

KUVIO 1. Esimerkki HTML-koodista

2.3 XML

XML tulee sanoista eXtensible Markup Language, ja se on suomennettuna laajennettu merkintäkieli. Kieli onkin joukko sääntöjä määrittelemään semanttisia merkintäkoodeja. XML on metamerkintäkieli, eli se määrittelee itse syntaksin, jota käytetään määrittelemään muita merkintäkieliä. (Harold 2000, 28.)

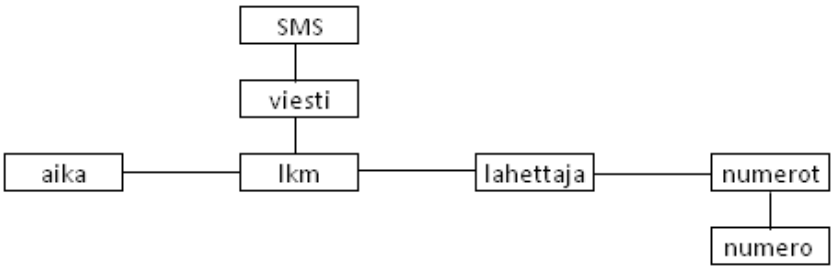
XML ei ole samanlainen merkintäkieli kuten esimerkiksi HTML. Mikäli HTML:ssä ei ole tiettyä merkintäkoodia, on ainoana vaihtoehtona odottaa kehittäjiltä uutta versiota, johon mahdollisesti sisällytetään kyseinen merkintäkoodi. XML:ssä käyttäjä voi itse luoda tarvitsemansa merkintäkoodit.

Merkit täytyy järjestellä tiettyjen periaatteiden mukaisesti, mutta merkitykseltään ne ovat hyvin joustavia. Tietoa ei myöskään ole pakotettu kappaleisiin, listan alkioihin, eikä tarvitse käyttää muita yleisiä luokkia. Kuviossa 2 on kuvitteellinen nelitasoinen XML-tiedosto, ja samankaltaista rakennetta hyödynnetään myös työssä. (Harold 2000, 28.)

```
1  <?xml version="1.0"?>
2  <kirjat>
3      <kirja>
4          <nimi>ABC-kirja</nimi>
5          <tekija>Mikael Agricola</tekija>
6          <vuosi>1543</vuosi>
7          <kappaleet>
8              <nimi>Johdanto</nimi>
9              <nimi>Alkeet</nimi>
10         </kappaleet>
11     </kirja>
12 </kirjat>
```

KUVIO 2. XML-tiedoston sisältö

XML:ää voidaan käyttää lokitiedoston kirjoittamiseen, mikäli asiakkaalla on tähän tarvetta. Lähetystietojen varastointiin on kuitenkin parempiakin tapoja, kuten tietokanta, joten XML-tiedoston kirjoittamista voidaan pitää varajärjestelmänä. XML-tiedon puumainen rakenne koostuu yhdestä juurielementistä SMS ja tämän alla olevasta viesti-elementistä. Tämä elementti pitää sisällään yhden lähetyskerran tiedot. Lähetyksestä otetaan talteen aika, lukumäärä, lähettäjä ja elementin numerot sisään tulee jokainen viestin saanut numero omana elementtinään, kuten kuviosta 3 näkyy.



KUVIO 3. XML-tiedoston rakenne

2.4 MySQL

MySQL on relaatiotietokantoja käsittelevä tietokannanhallintajärjestelmä. Tietokannat ovat tietovarastoja, joihin voidaan tallentaa tietoa ja joista voidaan hakea tietoa. Relaatiotietokannat ovat yksi tietokantatyyppi muiden joukossa ja nykyään ylivoimaisesti eniten käytetty tietokantatyyppi. Relaatiotietokannoissa tiedot on järjestetty tauluihin, ja taulujen väliset relaatiot eli suhteet ovat merkittävässä osassa. (Heinsuo & Ranta 2007, 38.)

MySQL:n on kehitellyt ruotsalainen MySQL AB, ja se olikin alun perin tarkoitettu yrityksen sisäiseen käyttöön. MySQL on suorituskykyinen, ja sitä voidaan käyttää useilla ohjelmointikielillä, kuten C-, C++ -, PHP-, Perl- ja Python-kielellä. Kuviossa 4 on kuvitteellinen MySQL-tietokantaan yhdistäminen PHP-kielellä ja taulujen tulostaminen. (Heinsuo & Ranta 2007, 38.)

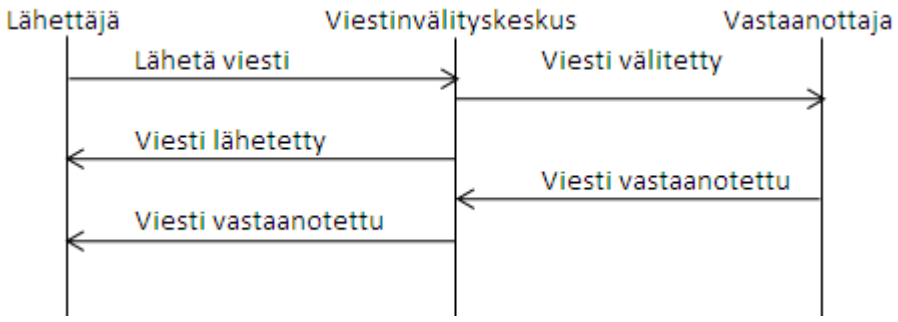
```php
<?php
$con = mysql_connect("localhost","root","root");
if (!$con)
  {
  die('Ei voitu yhdistää: ' . mysql_error());
  }

mysql_select_db("autot", $con);

$result = mysql_query("SELECT * FROM merkit");

while($row = mysql_fetch_array($result))
  {
  echo $row['mallit'];
  echo "<br />";
  }

mysql_close($con);
?>
```

KUVIO 4. MySQL-kantaan yhdistäminen ja rivien tulostus

3 MOBIILIVIESTINTÄ

Tekstiviesti on yksinkertaisten viestien siirtämiseen tarkoitettu lyhytviestipalvelu, toiselta nimeltään SMS (Short Message Service). Se otettiin ensimmäisen kerran käyttöön vuonna 1992. Tekstiviestejä käytetään tavallisesti puhelimesta toiseen kommunikointiin. Normaalisti käyttäjä kirjoittaa viestin omaan matkapuhelimeensa ja lähettää sen toiselle käyttäjälle käyttäen lähetysosoitteena toisen käyttäjän puhelinnumeroa. Viestinvälityspalvelu ottaa vastaan viestin, ja tämä toimiikin ns. store-and-forward yksikkönä, eli se varastoi viestin ja lähettää sen eteenpäin. Se myös vastaa viestin perillepääsystä. Viestinvälityspalvelu saattaa joutua odottamaan, että vastaanottajan puhelin tulee tavoitettavaksi (puhelin on päällä tai verkon alueella). Viestin vastaanotto ei vaadi mitään toimenpiteitä vastaanottajalta; viesti ilmestyy käyttäjän terminaaliin ja ilmoittaa käyttäjälle uuden viestin saapumisesta. (Nokia 2001, 163.)

Toinen tärkeä aspekti käytettävyyden kannalta SMS:ssä on, kun vastaanottaja on saanut viestin, lähetetään alkuperäiselle lähettäjälle viesti, että viesti on välitetty. Useiden ihmisten pitäessä puhelintaan mukana, on se kätevä tapa ilmoittaa, että viesti on mennyt perille. Kuvio 5 esittää tämän yksinkertaistetusti. (Nokia 2001, 163.)



KUVIO 5. Viestin lähettäminen

Toisaalta SMS sisältää useita rajoituksia. Jokainen SMS-viesti on rajoitettu maksimimerkkimäärään 160 merkkiä, vaikkakin on mahdollista linkittää useita viestejä yhteen. SMS-viestiin voidaan myös sisällyttää muunlaista binääritietoa, kuten kuvia ja soittoääniä, mutta koska SMS-viesti lähetetään signaalikanavia pitkin langattomassa verkossa, on se epäkäytännöllistä käyttää multimedian lähettämiseen viestien koon ollessa useita kilotavuja. (Nokia 2001, 163.)

SMS-viestit hyödyntävät signalointikanavien käyttämätöntä kapasiteettia ja tämän vuoksi niiden lähettäminen ja vastaanottaminen on mahdollista datan tai äänen lähetyksen aikana. Tänä päivänä SMS:ää käytetään esimerkiksi tietilanteiden, sähköpostiotsikoiden ja varastotilanteiden näyttämiseen (Schiller 2001, 85).

4 KÄYTTÖYMPÄRISTÖN KUVAUS

Itse tekstiviestin lähettämiseen käytetään Datapoli Oy:n ylläpitämää tekstiviestipalvelua, joka mahdollistaa 320 merkkisen viestin lähettämisen matkapuhelimeen. Palvelu toimii krediitti-periaatteella, jossa yksi 160 merkin viesti, joka on lähetetty yhdelle vastaanottajalle, merkitsee yhtä krediittiä. Palvelu kykenee myös Push-viesteihin, eli viesteihin, jotka tulevat suoraan matkapuhelimen näytölle ja häviävät kun, viestistä poistutaan, mutta näihin ei ole tarvetta tässä projektissa.

4.1 Tekstiviestipalvelu

Tekstiviestipalvelu sisältää luokkia, jotka mahdollistavat viestien lähetyksen ja lähetyksestä saatujen tietojen tarkastelun. SmsGateway-luokka lähettää SMS-viestit XML-muodossa HTTP POST-metodilla Medioso Pro yhdyskäytävälle, ja tämä välittää viestit vastaanottajille. SmsGatewaylla on kolme input-parametria, jotka ovat itse käytävän osoite, käytettävän tilin tunnus sekä salasana.

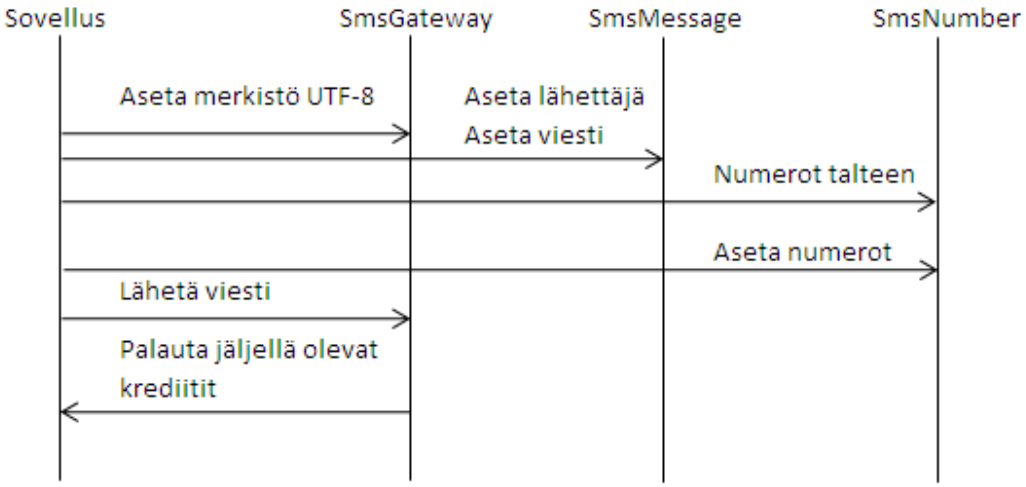
SmsGatewayException-luokka perii PHP:n oman Exception luokan, joka on kantaluokka kaikille poikkeuksille.

ResponseCodes-luokka on rajapintaluokka, joka sisältää viestin koodit virheen sattuessa. Kyseinen luokka implementoidaan SmsGatewayResponse-luokassa. SmsGatewayResponse-luokka sisältää metodeita, joilla voidaan muun muassa hakea palvelimelta tilin krediittien määrä, onnistuneesti lähetettyjen viestien määrä sekä epäonnistuneesti lähetettyjen viestien lukumäärä. Palvelimelta tuleva vastaus on XML-muodossa, ja se myös parsitaan SmsGatewayResponse-luokassa.

SmsMessage-luokassa määritellään kolme vakiomuuttujaa, jotka ovat viestin pituus sekä lähettäjän maksimi kirjainmäärä ja maksimi numeromäärä. Lähettäjän

nimi voi olla maksimissaan 11 merkkiä tai 14 numeroa. Tämä luokka myös asettaa itse viestin sisällön sekä viestin lähettäjän.

SmsNumber-luokassa varmistetaan, että numero on suomalainen GSM-numero. Kuviossa 6 esitellään yksinkertaistetusti, kuinka toiminnat suoritetaan eri luokkien välillä.



KUVIO 6. Sekvenssikaavio lähetykselle

4.2 Moodle

Työssä keskitytään Moodlen ympärille rakennettavaan pluginiin, joka mahdollistaa tekstiviestin lähetyksen Moodlen sisältä. Moodle on käytössä useimmissa korkeakouluissa kautta Suomen, ja tämän vuoksi lisäosan saaminen tuotteeksi mahdollistaa laajan asiakaskunnan.

Moodle muodostuu sanoista Modular Object-Oriented Dynamic Learning Environment. Moodle on Internet-pohjainen ohjelmisto kurssien ja web-sivujen tuottamiseen. Päätähtäimenä on tarjota vapaan lähdekoodin paketti koulutuksen helpottamiseksi ja interaktiivisen oppimisympäristön luomiseksi. Kehitys alkoi vuonna 1999, ja nykyinen versio 1.9.7 julkaistiin vuonna 2009. Päätekijänä toimii australialainen Martin Dougiamas. Moodle on pääosin toteutettu PHP-kielellä.

4.2.1 Ohjelmointikäytännöt

Moodle velvoittaa kehittäjiltään tietynlaista ohjelmointityyliä, mikä onkin kattavasti esiteltynä Moodlen dokumentoinnissa. Johdonmukainen ohjelmointi on tärkeää missä tahansa projektissa, ja sen edut tulevatkin esille, mikäli projektissa on useita kehittäjiä. Standardoitu tyyli helpottaa itse koodin lukemista ja ymmärtämistä, mikä taas parantaa projektin ylläpitoa. Moodlen kolme tärkeintä tavoitetta on yksinkertaisuus, luettavuus ja yhteensopivuus eri työkalujen kanssa. Tämä ilmenee metodien luomisessa, vakioissa ja kaavoissa, joita IDE-työkalut tukevat sekä metodien, luokkien ja vakioiden automaattisessa täytössä. (Moodle 2010.)

Lähdekoodin muotoilussa on seuraavia tapoja, joita kehittäjien tulisi noudattaa. PHP-tagien luonnissa käytetään aina pitkiä tageja eli PHP-koodi pyritään luomaan yhden tagin sisään eikä useisiin lyhyempiin tageihin, kuten kuviossa 7. Sisennyksessä käytetään neljää välilyöntiä ilman tabulaattoria. Tämän takia editorit tulisi asettaa ymmärtämään tabulaattori välilyönneiksi, jolloin vältytään syöttämästä uusia tabulaattorimerkkejä lähdekoodiin. Myöskään koodin ylintä tasoa ei sisennetä. (Moodle 2010.)

```
1  <?php
2      $muuttuja = 10;
3  |   | 4 välilyöntiä
4
5  ?>
6
7  <?php
8      $muuttuja2 = 'sanajoukko';
9  |   | 4 välilyöntiä
10
11 ?>
```

KUVIO 7. Useita lyhyitä PHP-tageja ja neljän välilyönnin sisennys

Rivien tavoiteltu maksimimerkkimäärä on 100 merkkiä. Tarkoituksena on pitää lähdekoodi helposti luettavana. Taulukoiden paketoinnissa rivinvaihtoa käytetään aina avain=>pari yhdistelmän ja pilkun jälkeen ja sisennyksessä käytetään kahdeksaa välilyöntiä. Mikäli funktion määrittelyssä on useita parametreja, sisennetään ne samalle tasolle ensimmäisen parametrin kanssa. Tehtäessä kontrolliraken teita, jotka sisältävät pitkiä ehtolauseita, on suositeltavaa käyttää muuttujia joillekkin niistä, koska se parantaa luettavuutta. (Moodle 2010.)

Tekstiformaattina käytetään standardia Unix-tekstiä. Rivit päättyvät rivinvaihtomerkkiin eli heksadesimaaleina 0x0A eikä rivinalkuunpalautusmerkkiin tai näiden yhdistelmiin, kuten Windows-tietokoneissa. Rivit eivät saa sisältää jälkivälilyöntejä, ja useimmat editorit voidaankin konfiguroida poistamaan nämä tallennuksen yhteydessä. Mikäli muotoillaan jo valmista Moodle-tiedostoa, ja tarkoituksena on lähettää tiedosto versionhallintaan, on yllä mainittu ominaisuus kytkettävä pois käytöstä, etteivät eroavat välilyöntimallit saastuta versionhallintaa. (Moodle 2010.)

Tiedostojen nimeämiskäytäntöjen mukaan tiedostonimien kuuluu olla kokonaisia englanninkielisiä sanoja, mahdollisimman lyhyitä, pienillä kirjaimilla kirjoitettuja ja päättyä päätteisiin .php, .html, .js, .css tai .xml. Luokkien nimet kirjoitetaan myös pienellä ja erotellaan alaviivalla. Vaikka luokka ei tarvitsekkaan input-parametrejä, käytetään funktion perässä funktiosulkeita. (Moodle 2010.)

Moodlessa suositellaan käytettävän PHP:n perusluokkaa stdClass, mikäli on tarvetta käyttää objektia ilman tiettyä luokkaa, kuten esimerkiksi valmisteltaessa tietoa syötettäväksi tietokantaan. stdClass on yleinen tyhjä luokka, jota käytetään kun tietotyyppejä muutetaan objekteiksi. Ennen Moodlen 2.0 versiota määriteltiin luokka, joka peri stdClass-luokan, ja luotiin siitä uusi objekti. Tämä ei ole enää käytössä. (Moodle 2010.)

Funktiot ja metodit tulee nimetä yksinkertaisesti ja käyttää pieniä kirjaimia. Moduulin nimi tulee olla ensin, jotta vältytään moduulien välisiltä ristiriidoilta, ja sanat tulee erotella alaviivoilla. Funktioiden nimien tarkoitus on olla kuvaavia käytännöllisyyden rajojen sisällä, sillä se edesauttaa koodin ymmärtämistä.

Funktion nimen ja funktiosulkeiden välissä ei käytetä välilyöntiä. Parametrit ovat aina yksinkertaisia englanninkielisiä sanoja, eikä niitä erotella alaviivalla. Null-arvoa käytetään false-arvon sijaan tilanteissa, joissa alkuarvoa ei tarvita. Mikäli valinnainen parametri on boolean-tyyppinen ja sen looginen oletusarvo on joko tosi tai epätosi, niin tällöin alkuarvo voi olla tosi tai epätosi. (Moodle 2010.)

Muuttujien nimissä käytetään helppolukuisia ja kuvaavia pienillä kirjaimilla kirjoitettuja englanninkielisiä sanoja. Jos tarvetta on käyttää enemmän kuin yhtä sanaa muuttujassa, tulee ne laittaa yhteen ilman alaviivaa ja pitää mahdollisimman lyhyinä. Monikkomuotoa käytetään taulukoista, joissa on objekteja. Globaaleja muuttujia kuvataan Moodlessa isoilla kirjaimilla, kuten esimerkiksi $CFG. Näitä ei tule luoda lisää. Vakiot nimetään aina isoilla kirjaimilla, ja ne alkavat moduulin nimellä. Erottelu tapahtuu alaviivoilla. Boolean-muuttujan arvot kirjoitetaan pienellä kuten myös null. (Moodle 2010.)

String-muuttujien suorituskyky ei ole ongelma PHP:n nykyisissä versioissa, joten pääpaino on luettavuuden edesauttamisella. Yksinkertaisia lainausmerkkejä käytetään, kun merkkijono on yksinkertainen tai sisältää paljon kaksoislainausmerkkjä. Kaksoislainausmerkit ovat vähemmän hyödyllisiä Moodlessa. Niitä käytetään, mikäli merkkijono sisältää muuttujia tai paljon yksinkertaisia lainausmerkkejä. Muuttujien käyttö merkkijonon sisällä voidaan suorittaa kahdella tavalla alla olevan kuvio 8:n mukaisesti.

```
1  $tervehdys = "Hei $kayttaja, mitä kuuluu?";
2
3  tai
4
5  $tervehdys = "Hei {$kayttaja}, mitä kuuluu?";
```

KUVIO 8. Muuttujien käyttö merkkijonoissa

Merkkijonot tulee yhdistää toisiinsa pisteoperaattorin avulla, kuten esimerkiksi $pitkastring = $useita.$lyhyita.’merkkijonoja’;. Mikäli rivit ovat pitkiä, katkaistaan ne useiksi riveiksi luettavuuden parantamiseksi aina pisteen jälkeen. Merkkijonot tulisi olla muotoa ”Suurin piirtein näin” eikä ”Ainakaan Näin”. Isoja

kirjaimia käytetään lauseen alussa tai kun käytetään nimeä, kuten Moodle. (Moodle 2010.)

Numeerisesti indeksoiduissa tauluissa negatiivisten indeksien käyttö on kiellettyä ja suosituksena on käyttää nollatta indeksiä ensimmäisenä indeksinä. Taulukon alustuksessa käytetään yhtä välilyöntiä jokaisen pilkun jälkeen luettavuuden parantamiseksi. Esimerkiksi $taulu = array(1, 2, 3);. Assosiatiivissa tauluissa suositellaan käytettävän useita rivejä, mikäli se parantaa luettavuutta. (Moodle 2010.)

Luokkien nimeämisessä käytetään Moodlen nimeämiskäytäntöjä. Funktion avaava aaltosulku tulee olla aina samalla rivillä funktion nimen kanssa, ja dokumenttiblokin tulee olla funktion yläpuolella, jonka tulee noudattaa PHPDocumentorin standardeja. PHPDocumentor luo valmiista PHP-lähdekoodista dokumentin, jossa on linkit esimerkiksi PHP:n omiin yleisiin dokumentteihin. Funktion sisällä oleva lähdekoodi sisennetään neljällä välilyönnillä. Ylimääräisen koodin sijoittaminen luokkatiedostoon on sallittua, mutta sitä ei suositella. Tällaisissa tiedostoissa jätetään kaksi tyhjää riviä, jotka erottavat lisätyt lähdekoodit. Jäsenmuuttujat tulee olla nimettynä Moodlen muuttujien nimeämiskäytäntöjen mukaan. Kaikki muuttujat luodaan ja alustetaan funktion alussa ennen metodeja. Var-muodostinfunktio ei ole sallittu. Jäsenmuuttujat alustetaan aina joko private-, protected- tai public-näkyvyydellä. Luvan antaminen muuttujille suoraan alustamalla se public-tyyppiseksi on sallittua, mutta set/get-apumetodien takia se ei ole suositeltavaa. (Moodle 2010.)

Funktiot tulee myös nimetä Moodlen nimeämiskäytäntöjen mukaan. Luokkien sisällä olevat metodit tulee aina alustaa näkyvyydeltään joko public-, private- tai protected-tyyppiseksi. Aivan kuten luokissa, myös funktioiden aloitussulku tulee olla samalla rivillä funktion nimen kanssa, eikä nimen ja sulun väliin tule välilyöntiä. Tämä saattaa haitata luettavuutta, minkä lisäksi se rikkoo koodia, mikäli metodia muutetaan myöhemmässä vaiheessa. Funktion argumentit erotellaan yhdellä välilyönnillä pilkun jälkeen. (Moodle 2010.)

If/else-rakenteissa käytetään yhtä välilyöntiä ennen ja jälkeen kontrollilausekkeen, ja samalla tavalla toteutetaan myös operaattorin erottelu. Sisennettyjä sulkuja käytetään, mikäli se parantaa loogista ryhmittelyä. Sisennys toteutetaan neljällä välilyönnillä. Sulkuja käytetään, vaikka lohko sisältäisi yhden rivin koodia. (Moodle 2010.)

Switch-rakenteessa käytetään välilyöntiä erottelemaan kontrollilauseke ja operaattorit. Sisennys joka valinnalle neljällä välilyönnillä ja sama sisennys jokaisen valinnan sisällölle. Samalla tavalla muotoillaan myös Foreach-rakenne. (Moodle 2010.)

Jokainen tiedosto tulee alkaa sisällyttämällä config.php-tiedosto, kuten esimerkiksi require_once(dirname(__FILE__) . '/../config.php');. Muut sisällytykset sisältävät absoluuttisen tiedostopolun alkaen $CFG->dirroot tai $CFG->libdir. Tämä esitellään kuviossa 9. Relatiiviset sisällytykset saattavat joskus toimia ennalta odottamattomasti, joten on turvallisempaa välttää niitä. (Moodle 2010.)

```
16      require_once($CFG->dirroot.'/user/profile/lib.php');
17      require_once($CFG->dirroot.'/tag/lib.php');
```

KUVIO 9. Tarvittavien config-tiedostojen sisällyttäminen

Moodle käyttää dokumentointiin phpDoc-formaattia. Tämä mahdollistaa dokumentoinnin automaattisen luonnin kyseisen työkalun avulla. Yhden rivin dokumentointi selittää lähdekoodia ja kertoo funktioiden sekä muuttujien tarkoituksen. Dokumentointia käytetään, kun se koetaan tarpeelliseksi. Yhden rivin dokumentointiin käytetään //-tagia sisennettynä samalle tasolle, kuin kyseisessä kohdassa oleva lähdekoodi on. (Moodle 2010.)

Jokaisen tiedoston, joka sisältää PHP-koodia, tulee sisältää GPL-tekijänoikeuslausunto tiedoston alussa sekä erillinen dokumentointilohko sen alapuolella. Tämä dokumentointilohko sisältää yhden rivin kokoisen selityksen tiedoston kuvauksesta, tämän jälkeen pidempi selostus tiedostosta, @package-tagi kuvaten, mikä osa Moodlea tämä tiedosto on, @copyright-tagi, jossa kerrotaan

miltä vuodelta tiedosto on ja tekijänoikeuksien haltijan sekä @license-tagin, joka sisältää ”http://www.gnu.org/copyleft/gpl.html GNU GPL v3 or later”. Kuvio 10 havainnollistaa PHP-tiedoston alun. (Moodle 2010.)

```php
<?php

// This file is part of Moodle - http://moodle.org/
//
// Moodle is free software: you can redistribute it and/or modify
// it under the terms of the GNU General Public License as published by
// the Free Software Foundation, either version 3 of the License, or
// (at your option) any later version.
//
// Moodle is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License
// along with Moodle.  If not, see <http://www.gnu.org/licenses/>.


/**
 * This is a one-line short description of the file
 *
 * You can have a rather longer description of the file as well,
 * if you like, and it can span multiple lines.
 *
 * @package    mod
 * @subpackage mymodule
 * @copyright  2008 Kim Bloggs
 * @license    http://www.gnu.org/copyleft/gpl.html GNU GPL v3 or later
 */
```

KUVIO 10. PHP-tiedoston alku

Package-tagi pitää olla sisällytettynä, ja se merkitsee tiedoston olevan osa jotain joukkoa tiedostoja. Tämän tagin käytännöt ovat seuraavat: 1. Mikäli tiedosto on osa Moodlen ydinkirjastoja, käytetään tagia @package moodlecore. Moodlen ydinkirjastojen sisällä voidaan käyttää tagia @subpackage, mikäli lähdekoodi kuuluu tiettyyn alueeseen, kuten esimerkiksi @subpackage questionengine. 2. Jos tiedosto on osa lisäosaa, käytetään lisäosan koko nimeä, kuten esimerkiksi @package mod_forum. 3. Testauslähdekoodi tulee olla sisällytettynä samaan pakettiin kuin testattava lähdekoodi. (Moodle 2010.)

Luokilla tulee olla dokumentointilohko, joka sisältää minimissään lyhyen kuvauksen luokasta, pidemmän kuvauksen luokasta(mikäli tarvetta), @copyright-tagin ja @license-tagin. (Moodle 2010.)

Kaikkien funktioiden ja metodien tulee sisältää kuvion 11 mukainen dokumenttilohko. (Moodle 2010.)

```
/**
 * The description should be first, with asterisks laid out exactly
 * like this example. If you want to refer to a another function,
 * do it like this: {@link clean_param()}. Then, add descriptions
 * for each parameter and the return value as follows.
 *
 * @param int   $postid The PHP type is followed by the variable name
 * @param array $scale The PHP type is followed by the variable name
 * @param array $ratings The PHP type is followed by the variable name
 * @return mixed
 */
```

KUVIO 11. Funktioiden dokumenttilohko

Lisättäessä uusia luokkia tai funktioita Moodlen ydinkirjastoihin (tai lisättäessä uusi metodi olemassa olevaan luokkaan) käytetään @since-tagia kertomaan, mille Moodle-versiolle se lisättiin. Vanha API merkitään vanhentuneeksi @deprecated-tagia käyttämällä. Mikäli on tärkeää, että kehittäjät päivittävät lähdekoodia, voidaan käyttää testauskutsua (’..’, DEBUG_DEVELOPER);, jolloin saadaan kaikki pienimmätkin virheilmoitukset tulostettua kehittäjille. (Moodle 2010.)

TODO-kommenttien tulee olla yhdistettynä Moodlen tietokantaan, joka pitää sisällään ohjelmistovirheet, parannukset ja ominaisuudet. TODO-kommentti tehdään seuraavasti: ’//TODO MDL-456 tarvitaan parempi algoritmi’ ja näin merkitään keskeneräiset osat koodista. Tietokannasta ongelmaa ei voida poistaa, mikäli koodissa on jäljellä TODO-tagi. (Moodle 2010.)

Poikkeuksia käytetään raportoimaan virheitä, varsinkin kirjastojen lähdekoodissa. Poikkeuksen heittäminen on lähes sama asia kuin print_error:n kutsuminen, mutta se on paljon joustavampi. Mikä tahansa poikkeus, jota ei oteta kiinni, laukaisee kutsun print_error:lle, ilmoittaen ongelmasta käyttäjälle. Poikkeuksia ei tule

hyväksikäyttää normaalissa lähdekoodissa, vaan niitä tulee käyttää ainoastaan vikaherkissä tilanteissa. (Moodle 2010.)

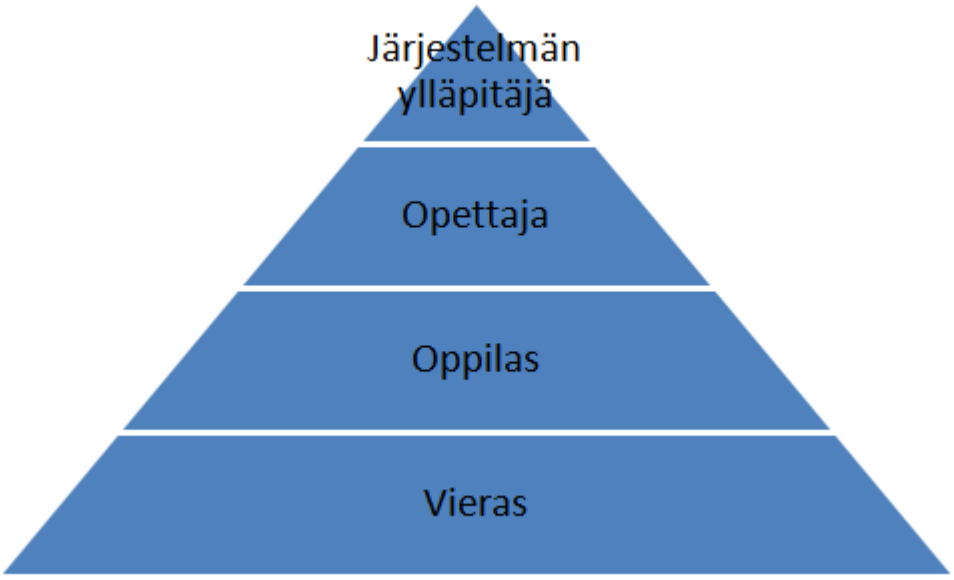
PHP sisältää useita kyseenalaisia toimintoja, joiden käyttö on erittäin epäsuotavaa, koska ne ovat usein syynä turvallisuusongelmiin. Seuraavassa on lista kyseisistä toiminnoista:

- 1. Älä käytä eval()-funktiota, kielipaketit ovat poikkeus tähän ja tämä tullaan korjaamaan tulevaisuudessa.
- 2. Älä käytä preg_replace() /e-argumentilla vaan käytä takaisinkutsuja ei-toivotun PHP-koodin ajon estämiseksi.
- 3. Älä käytä `-merkkiä komentorivi komentojen ajamiseen. (Moodle 2010.)

4.2.2 Käyttäjien roolit

Koska Moodle pitää sisällään käyttäjien tietoja ja esimerkiksi oppilaiden kurssiarviointeja, on erittäin tärkeää pitää ympäristö tietoturvallisena. Tarve tietoturvalle kasvaa myös sen takia, koska Moodle on Internet-pohjainen oppimisympäristö, jolloin altistuminen hyökkäyksille on todennäköisempää kuin paikallisesti toimivassa ympäristössä. Internet-pohjaisen sovelluksen turvallisuus riippuukin vahvasti siitä, miten eri rooleja käytetään käyttäjien turvaamiseksi joko itseltään tai ulkopuoliselta uhalta. (Moodle 2010.)

Rooleja Moodlessa on neljä kappaletta, järjestelmän ylläpitäjä, opettaja, oppilas ja vieras(KUVIO 12.) Järjestelmän ylläpitäjällä on oikeus muuttaa kaikki asetuksia, luoda kursseja, tarkastella kursseja, muokata kielipaketteja ja muokata kaikkia käyttäjiä. Epäsuorasti järjestelmän ylläpitäjä voi ajaa komentorivi- tai PHP-koodia. Moodlen järjestelmän ylläpitäjä eli admin voi olla osittain rajoitettu kovakoodaamaan asetuksia config.php-tiedostoon. Pohjatason admineita ei pystytä rajoittamaan, koska he pääsevät käsiksi PHP-tiedostoihin ja voivat muuttaa niitä. Luonnollisesti kaikkien adminien tulee olla luotettavia. (Moodle 2010.)



KUVIO 12. Roolit Moodlessa

Opettajien roolina on useimmiten luoda kurssin sisältöä, hyväksyä oppilaat kurssille ja itse opettaminen. Kyseiselle roolille onkin annettu seuraavat oikeudet: Ladata tiedostoja palvelimelle, lähettää HTML-tekstejä, luoda aktiviteetteja ja hallinta sekä pääsy oppilaiden arvosanoihin ja muuhun henkilökohtaiseen tietoon. JavaScriptin, Flashin tai muun skriptatun sisällön lataaminen on usein katsottu olevan riski tietoturvalle. Näihin asioihin Moodlen on vaikea vaikuttaa, sillä edellä mainittua sisältöä käytetään myös normaalissa opettamisessa. Selain ei tiedä, millaista sisältöä kurssi sisältää tai millaiselta palvelimelta kyseinen sisältö tulee. Tiedostosta tulee osa palvelinta, kun se on sinne tallennettu. Kaikkien opettajien, jotka ovat saaneet riskialttiita valtuuksia, tulee olla luotettavia, eikä opettajan roolia voida antaa opiskelijalle. (Moodle 2010.)

Opiskelijat ottavat osaa kursseihin eikä heihin voi luottaa. Opiskelijat tarvitsevat seuraavat oikeudet: Julkaista muotoiltua tekstiä, joka sisältää kuvia ja liitteitä. Ladattuja tiedostoja ei tule suoraan saada avata selaimella samalta palvelimelta. Tämän sijaan tiedostot tulee jakaa toisesta domainista tai palvelimen tulee ladata kaikki tiedostot pakotetusti paikalliselle asemalle ennen avausta. Kaikki opiskelijoiden lähettämä teksti tulee puhdistaa XSS-hyökkäysten tai Flashin, JavaScriptin, vektorigrafiikan sekä HTML:n sisällä olevien ohjelmien ajon

estämiseksi. XSS-hyökkäys tarkoittaa sitä, että hyökkääjä saa tavalla tai toisella syötettyä sivustolle koodia, joka ei sinne kuulu. HTML-koodin puhdistamiseen voidaan käyttää mustamaalausta eli poistetaan tarpeettomat elementit koodista tai HTML Purifier:ia, joka tekee edellä mainitun poistamisen, mutta lisäksi muokkaa sisällön standardien mukaiseksi. (Moodle 2010.)

Vieraille ei turvallisuussyistä anneta lupaa ladata palvelimelle tiedostoja eikä lähettää tekstiä, jota säilytettäisiin palvelimella. Vieraan roolin saaneet voivat tietysti häiritä muita käyttäjiä, hyväksikäyttää ongelmia HTML-koodin puhdistusrutiineissa, hyväksikäyttää muita haavoittuvuuksia tai tekeytyä esimerkiksi järjestelmän ylläpitäjäksi ja tätä kautta hankkia tietoonsa käyttäjän tietoja. Sivustot, joissa käytetään sähköpostipohjaista rekisteröintiä, joutuvat käyttämään enemmän huolellisuutta spam-viestien poistoon ja muunlaisiin hyökkäyksiin. (Moodle 2010.)

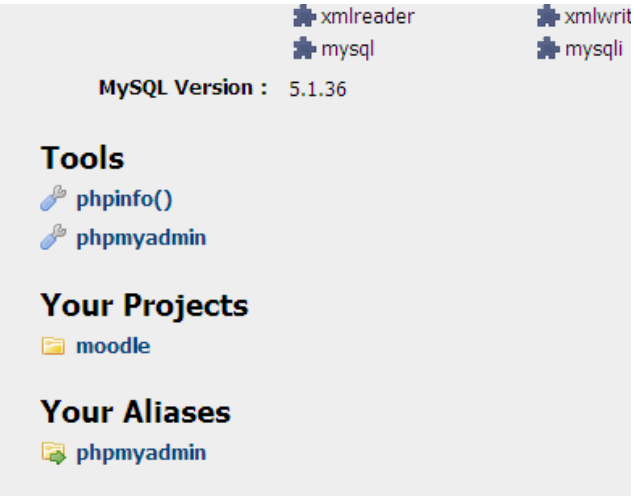
4.2.3 Asennus

Projektin kehitysvaiheessa Moodle asennettiin yhdelle työasemalle testausta varten. Moodle vaatii toimiakseen PHP:n lisäksi tietokannan ja Apache-serverin. Linuxille kyseisen paketin saa lataamalla esimerkiksi LAMP-paketin (Linux, Apache, MySQL, PHP). Mutta koska työasema, jolle Moodle asennettiin, oli Windows-käyttöjärjestelmällä varustettu, oli työasemalle ladattava Windows-ympäristön vastaava ohjelmisto. Opintojen aikana on käytetty WAMP:ia (Windows, Apache, MySQL, PHP), ja tämän paketin asentamiseen myös päädyttiin.

WAMP:n asennuksen jälkeen pitää suorittaa muutamia toimenpiteitä, ennen kuin ohjelmistoa voidaan käyttää. Ensin on luotava tyhjä tietokanta, johon Moodle tallentaa kaiken tiedon. WAMP sisältää myös PHPMyAdmin-ohjelman, joilla voidaan hallita ja muokata tietokantoja. Moodlen tietokannan tulee olla nimeä moodle ja samalla tulee luoda admin-käyttäjä kantaan, jolla on kaikki oikeudet kannan käsittelyyn. Moodle luo myös itselleen moodledata-nimisen kansion, johon se tallentaa kaiken ladatun sisällön, kuten kurssien dokumentit ja käyttäjien

kuvat. Tämän Moodle luo asennuksen aikana, mutta mikäli se jostain syystä ei saa sitä luotua, on käyttäjän itse luotava se.

Moodlen oma hakemisto sijoitetaan WAMP:n www-kansion juureen, ja sinne pääsee käsiksi joko kirjoittamalla Internet-selaimeen http://localhost/ tai navigoimalla WAMP:n tehtäväalueen pikakuvakkeesta Localhost-kohtaan. Tällöin avautuu näkymä paikallisen serverin tietoihin, joista näkyvät muun muassa Apachen, MySQL:n ja PHP:n versionumerot sekä asennetut lisäosat. Lisäksi näkyy myös kaikki käyttäjän projektit, joita www-kansioon on sijoitettu.



KUVIO 13. WAMP:n näkymä käyttäjän omista projekteista

Näiden toimenpiteiden jälkeen Moodle on valmiina asennusta varten. Moodlen asennus voidaan käynnistää kahdella tapaa: Ajamalla install.php Moodlen hakemiston juuresta tai mikäli Moodlen versio on 2.0 tai uudempi, voidaan se asentaa myös komentoriviltä. Asennus käynnistetään syöttämällä selaimeen http://localhost/moodle/install.php ja tämän jälkeen Moodle käy läpi serverin asetuksia ja ehdottaa parannuksia, mikäli ongelmia esiintyy. Samalla asennus myös luo tarvittavat taulut Moodlen tietokantaan käyttäjiä, kursseja ja muita asetuksia varten. Asennuksen edettyä loppuun on Moodle käyttövalmis.

4.2.4 Tietokanta

Moodle tallentaa tietonsa tietokantoihin, joiden tarkastelu olikin työn tekovaiheessa erittäin tärkeää. Moodle luo valittuun kantaan oman moodle-tietokannan, johon se tallentaa kaiken tiedon. Tietokanta sisältää noin 200 taulua. Tietokannan taulut on jaettu useaan osa-alueeseen, jotka ovat Configuration (Asetukset), Users and their profiles (Käyttäjätiedot), The roles and capabilities system (Roolit), Courses and their organisation into categories (Kurssit), Groups and groupings (Ryhmät), The logging system (Tiedon kirjaaminen ylös), Blocks system (Blokit eli Moodlen sisäiset sovellukset), Events (Tapahtumat ja niiden käsittely), Backup and restore (Varmuuskopiointi ja palautus), Statistics (Statistiikka), Tags (Tagit), Gradebook (Arvosanat), Messaging system (Viestinvälitys), Moodle Network (Moodle-verkko), Caching (Varastointi) ja Activity modules (Aktiviteetit). (Moodle.org)

Roolit-taulun alkiot sisältävät neljä määritelmää: roolin, kyky (capability), lupa (permission) ja konteksti(context). Roolilla määritellään käyttäjän asema jossakin kontekstissa, esimerkiksi opettajalla ja oppilaalla on eritasoisia rooleja. Kyvyllä tarkoitetaan Moodlen jotain ominaisuutta, kuten foorumilla vastauksen tekeminen mod/forum:replypost. Luvalla tarkoitetaan jossain ominaisuudessa annettua roolia, joita ovat esimerkiksi salli tai estä. Kontekstilla tarkoitetaan jotain tiettyä osaa Moodlen sisällä, esimerkiksi kursseja, blokkeja tai aktiviteettimoduleita. (Moodle 2010.)

Blokit eli lohkot ovat Moodlen kursseihin lisättäviä lisäosia, joita opettaja-roolin saanut käyttäjä voi lisätä kurssiinsa. Lohkot lisätään kurssille vetovalikosta, mikäli järjestelmän ylläpitäjä ei ole lohkoa pakottanut suoraan kurssin toteutukseen.

```php
<?php
class block_simplehtml extends block_base {
  function init() {
    $this->title   = get_string('simplehtml', 'block_simplehtml');
    $this->version = 2004111200;
  }
```

KUVIO 14. Lohkon määritys

Lohkon luominen aloitetaan luomalla Moodlen juureen blocks-kansioon halutun lohkon niminen kansio, esimerkiksi blocks/SMS. Kansion sisään luodaan block_SMS.php-niminen tiedosto, joka sisältää lohkon lähdekoodin. Lohkon määrittely tehdään kuvion 14 mukaisesti. Init()-funktion avulla määritellään sen sisällä olevat kaksi jäsenmuuttujaa. $this->title määrittelee lohkon nimen, joka näkyy ylätunnisteessa. Kuviossa 14 käytetään lohkon omaa kielitiedostoa, josta nimi luetaan. $this->version kohdassa määritellään lohkon versionumero. (Moodle 2010.)

```php
function get_content() {
    if ($this->content !== NULL) {
      return $this->content;
    }

    $this->content         =  new stdClass;
    $this->content->text   = 'The content of our SimpleHTML block!';
    $this->content->footer = 'Footer here...';

    return $this->content;
  }
}   // Here's the closing curly bracket for the class definition
    // and here's the closing PHP tag from the section above.
?>
```

KUVIO 15. Lohkon sisältö

Kuviossa 15 luodaan lohkolle sisältöä. Mikäli sisältöä ei haluta näyttää, muutetaan $this->content = NULL. Lohkolle asetetaan sisältöä $this->content->text:llä ja alatunniste $this->content->footer:lla. Mikäli lohkolle halutaan antaa jonkin näköisiä editointimahdollisuuksia, onnistuu se helposti asettamalla funktiolle instance_allow_config():lle palautusarvo true. Tällöin voidaan esimerkiksi antaa opettajalle lupa muokata esimerkiksi lohkon tekstiä. Mikäli lohkossa halutaan käyttää tietokantoja, luodaan SMS-kansion sisään db-kansio ja tämän sisään install.xml ja tämän sisään määritellään taulun skeema. Tämä tehdään

kuvitteellisesti kuviossa 16. XML-tiedostossa määritellään eri kenttien nimet, tyypit, koot ja muut ominaisuudet, joita tarvitaan taulun toimintaan. Samalla määritellään myös primääriavain, jolla yleensä määritellään uniikki kolumni,

```
<?xml version="1.0" encoding="UTF-8" ?>
<XMLDB PATH="blocks/SMS/db" VERSION="20060912" COMMENT="XMLDB file for Moodle SMS block"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="../../../lib/xmldb/xmldb.xsd"
>
  <TABLES>
    <TABLE NAME="block_SMS" COMMENT="SMS block for Moodle for sending SMS messages.">
      <FIELDS>
        <FIELD NAME="timesent" TYPE="bigint" LENGTH="10" NOTNULL="true" UNSIGNED="true" S
        <FIELD NAME="count" TYPE="bigint" LENGTH="10" NOTNULL="true" UNSIGNED="true" DEFA
        <FIELD NAME="userid" TYPE="int" LENGTH="small" NOTNULL="true" SEQUENCE="false" EN
        <FIELD NAME="courseid" TYPE="int" LENGTH="10" NOTNULL="true" SEQUENCE="false" ENU
    </FIELDS>
      <KEYS>
        <KEY NAME="primary" TYPE="primary" FIELDS="id" />
      </KEYS>
    </TABLE>
  </TABLES>
</XMLDB>
```

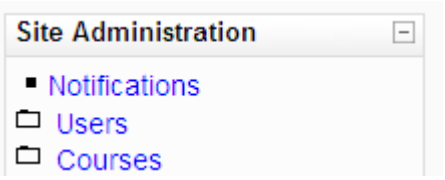
kuten esimerkiksi käyttäjien sosisaaliturvatunnukset. (Moodle 2010.)

KUVIO 16. Lohkon taulun luonti

Pääasiassa työssä käytetään lähinnä User-, Block-, Group- ja Course-tauluja. Tietyille käyttäjäryhmille annetaan myös oikeus lähettää tekstiviesti, ja nämä oikeudet annettiin Teacher-roolille eli opettajille.

4.2.5 Lisäosien asennus

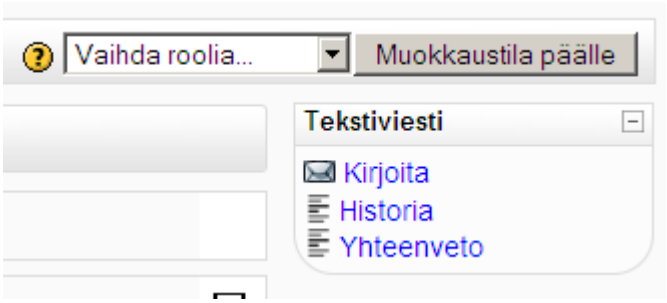
Moodlessa lisäosista käytetään nimitystä blocks. Asentaminen on lähestulkoon identtistä Moodlen itsensä asentamiseen, eli blockit puretaan kansioon moodle/blocks/ ja tämän jälkeen kirjaudutaan Moodleen. Asennus tapahtuu valitsemalla Notifications-kohta navigaatiomenusta, joka näkyy kuviossa 16, ja uusi lisäosa tunnistuu automaattisesti. Moodle asentaa lisäosan, mikäli käyttäjä hyväksyy sen.



KUVIO 16. Notifications-kohta navigaatiomenussa

5 TEKSTIVIESTILISÄOSA

Lisäosan tarkoituksena on, että opettajat voivat lähettää esimerkiksi äkillisissä sairaustapauksissa kurssin oppilaille tekstiviesti-ilmoituksia. Kuviossa 17 näkyy opettajan näkymä Moodlen kurssin sisällä. Kirjoita-linkistä pääsee kirjoittamaan tekstiviestiä ja Historia-kohdasta pääsee näkemään omat lähetetyt viestit. Yhteenveto-linkki näkyy vain järjestelmän ylläpitäjälle, ja sieltä näkyy kaikkien viestejä lähettäneiden opettajien lähettämät viestimäärät kursseittain.

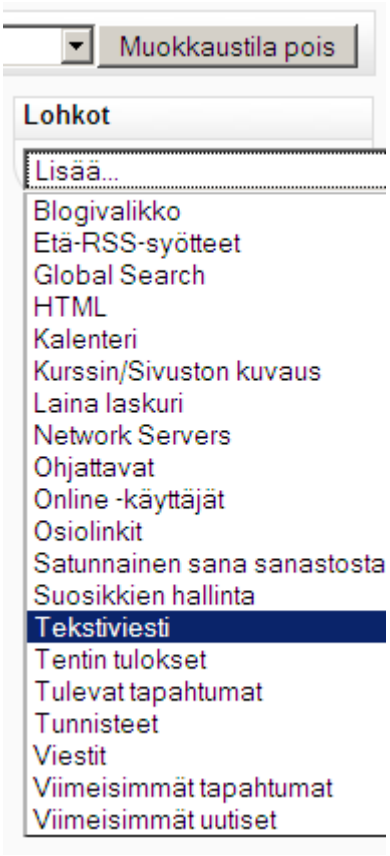


KUVIO 17. Kurssinäkymä

Jo toiminnassa olevista lisäosista saatiin apua käyttäjien valinnan toteutuksessa sekä lähetyssivun ulkoasussa. Muuten toiminnallisuudet tehtiin itse. Eri ominaisuuksista ja parannusehdotuksista keskusteltiin pilotissa olleiden henkilöiden kanssa ja pohdittiin, miten lisäosa olisi mahdollisimman yksinkertainen ja informoiva niin opettajan kuin oppilaankin kannalta.

Lohkon lisääminen halutulle kurssille on yksinkertaista. Opettaja asettaa oikealta ylhäältä muokkaustilan päälle ja valitsee alasvetovalikosta halutun lohkon. Valinnan jälkeen lohko ilmestyy kurssin oikeaan laitaan ja pysyy siellä, kunnes opettaja sen itse poistaa. Lohkon poisto tapahtuu lähes samalla tavalla eli asetetaan muokkaustila jälleen päälle ja painetaan halutun lohkon sisältä mustaa

ristiä. Tällöin lohko poistuu kurssin sisältä. Lohkon poistaminen kurssilta ei hävitä lohkoa, vaan sen voi lisätä kurssille, kun tarvetta taas on.

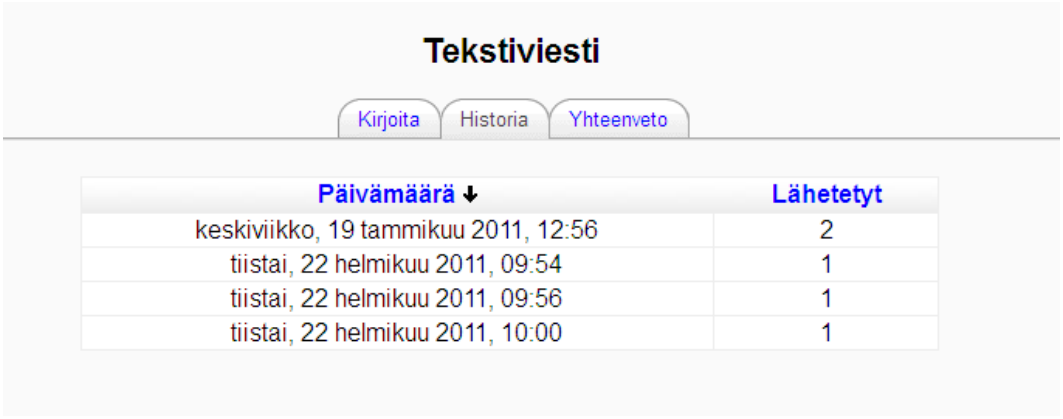


KUVIO 18. Lohkon lisääminen kurssille

Tekstiviestin lähetys käsittää vastaanottajien valinnan, viestikentän ja Lähetä- ja Peruuta-näppäimet. Peruuta-näppäimellä päästään takaisin valittuun kurssiin ja Lähetä-näppäin uudelleenohjaa myös takaisin kurssille lähetyksen jälkeen. Viestikentän pituus on rajoitettu JavaScriptillä 160:n merkkiin. Skripti laskee painetut näppäimet ja vähentää sen arvosta 160. Se myös osaa käsitellä kopioi- ja liitä-komennot. Liitteessä 3 näkyy koko lähetyssivu sisältäen Historia- ja Yhteenveto-välilehden.

Tekstiviestien kulun seuraamista varten lisäosassa on käyttäjän omia lähetettyjä viestejä seuraava sivu, joka näkyy kuviossa 19. Tällä sivulla käyttäjä näkee omat lähetetyt viestinsä päivämäärän mukaan lajiteltuna kappalemäärien kanssa. Järjestelmänvalvojia varten on vielä yksi sivu, jossa näkyvät kaikkien käyttäjien lähettämät viestit puolen vuoden sisällä kursseittain ja käyttäjien mukaan

lajiteltuna. Tämä sivu on esitetty kuviossa 20. Tämä sivu ei näy lisäosan sisällä kenelläkään muilla kuin sellaisilla henkilöillä, joiden roolina on järjestelmänvalvoja.



KUVIO 19. Historia-välilehti

KUVIO 20. Yhteenveto-välilehti

5.1 Sisältö

Moodlen kattavan lisäosatarjonnan ansiosta projektia ei tarvinnut aloittaa täysin tyhjältä pöydältä. Pohjaksi valittiin lisäosa, joskin toimimaton, jonka tarkoituksena oli lähettää sähköpostiviestejä kurssin sisällä valituille henkilöille. Lisäosasta pystyttiin käyttämään hyväksi käyttäjien valinta ja tarvittavien ryhmien luonti sekä osa lähetysformia.

Itse formia jouduttiin muokkaamaan joltain osin, sillä se oli tarkoitettu sähköpostia varten. Lisäksi lisättiin tekstiviestiä varten näppäinlaskuri. Lähetysskriptin osalta muutoksia tuli hieman enemmän. Puhelinnumeroa varten tehtiin niin kutsuttu säännöllinen lauseke – tarkistus (englanniksi regular expression), jolla tarkistetaan, että käyttäjän antama puhelinnumero on oikeaa muotoa eli maatunnus ensin ja numero sen jälkeen ilman välilyöntejä, kuten esimerkiksi +358441112222. Mikäli numero on eri muotoa, ei kyseistä lähettäjää voida valita vastaanottajaksi tekstiviestille. Numero varastoidaan tietokantaan phone1-nimikkeellä. Validate-funktio ottaa numeron input-parametrina ja palauttaa joko truen tai falsen riippuen siitä, onko numero oikeaa muotoa vai ei. Itse funktio, joka tämän tarkistaa katsoo, että numero alkaa joko +358 tai 0 numerolla. Tämän jälkeen varmistetaan, että seuraava numero on joko 4 tai 5. Seuraavan numeron on oltava väliltä 0-6, ja sen jälkeen numerossa voi olla 6-10 numeroa väliltä 0-9. Kuviossa 21 on kyseinen funktio koodissa.

```
6  function validate($number)
7  {
8      return preg_match('/^(?:\+?358\s*|0)(?:4|5)[0-6]{1}(?:\s*\d){6,10}$/',$number);
9  }
0
```

KUVIO 21. Numeron validointi

Itse lähetysfunktioon otetaan input-parametrina $_POST -funktiossa kaikki formilta tuleva tieto muuttujaan $temp sekä kurssin id muuttujaan $kurssi. Formilta tulevista tiedoista käytetään hyväksi viestikentän sisältö sekä käyttäjien puhelinnumerot. Nämä tiedot välitetään eteenpäin lähetysfunktiolle.

Lähetysfunktiossa otetaan $temp-muuttujasta lähetettävä viesti muuttujaan $viesti ja lähettäjä muuttujaan $lahettaja. Tämän jälkeen Medioso_SmsGateway luokasta luodaan uusi olio $gateway, johon sijoitetaan yhdyskäytävän osoite, käyttäjätunnus ja salasana. Yhdyskäytävän osoite on http://medioso.net/send. Tämän jälkeen asetetaan tekstin muoto UTF-8:ksi, jotta skandinaaviset merkit saadaan toimimaan oikein. Medioso_SmsMessagesta luodaan uusi ilmentymä $MediosoMessage ja tälle välitetään $viesti-muuttuja ja samalla otetaan setSender-funktioon lähettäjä $lähettäjä-muuttujasta.

Vastaanottajien numerot sijoitetaan $numerot-nimiseen tauluun, joka on Medioso_SmsNumber luokan ilmentymä, ja tätä iteroidaan, kunnes kaikki vastaanottajien numerot on käyty läpi. addNumbers-funktioon, joka on Medioso_SmsMessage-luokan funktio, syötetään $numerot-taulun sisältö, jolloin kaikki numerot saadaan talteen lähetystä varten. Viesti lisätään addMessage()-funktion sisääntulevana parametrina, jonka luokka on Medioso_SmsGateway. Itse lähetys on Medioso_SmsGateway luokan funktio send() ja lähetyksestä paluuarvona otetaan palvelimelta tuleva vastaus muuttujaan $response. $response-muutujasta voidaan nähdä, mikäli lähetys menee läpi onnistuneesti tai tapahtuu joku virhe. Palvelimelta tulevasta vastauksesta otetaan getSended()-funktiolla talteen onnistuneesti lähetettyjen viestien määrä.

5.2 Lähetystietojen varastointi

Viestien lähetyksen seurantaa varten lisäosaan sisällytettiin XML-tiedoston kirjoitusfunktio, joka tallentaa lähetyksestä ajan, viestien lukumäärän, lähettäjän sekä numerot, joihin viesti lähetettiin. Tiedostoon kirjoitetaan vain, mikäli lähetys on onnistunut ja se luodaan, mikäli sitä ei tallennushetkellä löydy.

Aika haetaan PHP:n date-funktiolla, ja formaattina on ISO 8601, joka lisättiin PHP:n viidenteen versioon. Viestien lukumäärää varten käydään numerot-muuttuja läpi count-funktion avulla, joka laskee taulukon kaikki elementit. Lähettäjän tiedot otetaan Moodlen sisäänkirjautumistiedoista globaalin USER-muuttujan avulla, esimerkiksi $USER->firstname ja $USER->lastname. Numeroiden tallennusta varten käydään numerot-taulu läpi uudelleen, ja joka kerralla kun löytyy uusi numero, kirjoitetaan se <numero>-tagin sisälle. XML-tiedoston rakenne näkyy kuviosta 22 ja luontia varten koodi kuviosta 23.

```
<?xml version="1.0" ?>
- <SMS>
  - <viesti>
      <aika>2010-04-01T07:50:09+00:00;</aika>
      <lkm>1</lkm>
      <lahettaja>Admin Käyttäjä</lahettaja>
    - <numerot>
        <numero>+358445550884</numero>
      </numerot>
    </viesti>
```

```
if($response->success())
{
    if(!file_exists("tiedot.xml"))
    {
        $xml_filu = fopen( "tiedot.xml", "a" );
        fwrite($xml_filu, "<?xml version='1.0'?>");
        fwrite($xml_filu, "<SMS>");
        fwrite($xml_filu, "<viesti>");
        fwrite($xml_filu, "<aika>".date('c').";</aika>");
        fwrite($xml_filu, "<lkm>".count( $numerot )."</lkm>");
        fwrite($xml_filu, "<lahettaja>".$kayttaja."</lahettaja>");
        fwrite($xml_filu, "<numerot>");
        for($i = 0; $i < count($numerot); $i++)
        {
            fwrite($xml_filu, "<numero>".$numerot[$i]->getNumber()."</numero>");
        }
        fwrite($xml_filu, "</numerot>");
        fwrite($xml_filu, "</viesti>");
        fwrite($xml_filu, "</SMS>");
        fclose($xml_filu);
    }
```

KUVIO 22. XML-tiedosto

KUVIO 23. XML-tiedoston kirjoitus

Projektin edetessä pohdittiin palaverissa, että lähetystietojen varastointi XML-tiedostona on liian riskialtis, ja näin päädyttiinkin tekemään pieniä muutoksia tietojen varastointiin. Luonnollinen vaihtoehto onkin laittaa tiedot suoraan tietokantaan.

Moodlen kantaan saadaan automaattisesti luotua taulu kirjoittamalla XML-tiedosto, johon laitetaan tarvittavat tiedot taulusta ja sen alkioista, kuten kuviossa x näkyy. Tämä sijoitetaan lohkon db-kansion sisälle, josta Moodle ymmärtää etsiä kyseistä tiedostoa lisäosan asennuksen jälkeen. Viestien tallennukseen käytetään mdl_block_tekstiviesti_log-taulua. Viesteistä varastoidaan neljä tietoa, jotka ovat käyttäjän id, kurssin id, lähetettyjen viestien määrä ja päivämäärä.

```
<?xml version="1.0" encoding="UTF-8" ?>
<XMLDB PATH="blocks/quickmailSMS/db" VERSION="20071126" COMMENT="XMLDB fil
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="../../../lib/xmldb/xmldb.xsd"
>
  <TABLES>
    <TABLE NAME="block_Tekstiviesti_log" COMMENT="Stores the message his
      <FIELDS>
        <FIELD NAME="count" TYPE="int" LENGTH="10" NOTNULL="true" UNSIGN
        <FIELD NAME="courseid" TYPE="int" LENGTH="10" NOTNULL="true" UNS
        <FIELD NAME="userid" TYPE="int" LENGTH="10" NOTNULL="true" UNSIG
        <FIELD NAME="timesent" TYPE="int" LENGTH="10" NOTNULL="true" UNS
      </FIELDS>
      <KEYS>
        <KEY NAME="courseid" TYPE="foreign" FIELDS="courseid" REFTABLE="
        <KEY NAME="userid" TYPE="foreign" FIELDS="userid" REFTABLE="user
      </KEYS>
    </TABLE>
  </TABLES>
</XMLDB>
```

KUVIO 24. Tietokantataulun luonti

6 TEKSTIVIESTIT OSANA VIESTINTÄÄ

Tekstiviesteille organisaatioiden sisällä on tällä hetkellä muutakin käyttö kuin pelkästään poissaoloilmoitusten antaminen. Helsingissä Laurea-ammattikorkeakoulu käyttää tuntiperuutuksia varten tekstiviestejä, mutta käytössä on myös hätätekstiviestipalvelu. Tarve kyseiselle palvelulle on syntynyt viimeaikaisten kouluammuskelutapauksien johdosta (Yle 2010).

Myös Hämeenkoskella on käytössä vastaava järjestelmä. Paikallisella ala-asteella järjestelmä toimii niin, että matkapuhelimella otetaan yhteys Internet-sivulle, jossa on käyttäjätunnukset. Viestejä varten on käytössä viestipohjia, joihin täytetään tarvittava muuttuva tieto, ja samalla valitaan, lähetetäänkö tieto kaikille oppilaiden vanhemmille vai vain osalle heistä. Siitä tieto lähtee valittujen vanhempien matkapuhelimiin. Tarkoituksena on käyttää kuitenkin palvelua vain kaikkein kriittisimpään viestintään, ei normaaliin koulun ja kodin väliseen tiedottamiseen.

Yleisellä tasolla tekstiviestit ovat sujuva kommunikointikeino varsinkin nuorten keskuudessa. Yksilölle tekstiviestistä on useitakin etuja, koska se on suhteellisen halpaa eikä se ole liian tunkeileva tapa kommunikoida (Nokia 2001, 149).

Vuosi	Lähtevät tekstiviestit, 1000 kpl [1)]	Muutos, %	Tekstiviestit/ liittymä, kpl	Lähtevät multimedia-viestit, 1000 kpl	Muutos, %
2002	1 324 668		293		
2003	1 647 218	24,3	347	2 314	
2004	2 193 498	33,2	439	7 386	219,2
2005	2 728 230	24,4	507	15 993	116,5
2006	3 087 998	13,2	544	21 568	34,9
2007	3 182 362	3,1	524	28 682	33,0
2008	3 566 523	12,1	517	37 801	31,8

KUVIO 25. Tekstiviestien lukumäärät vuosittain (Tilastokeskus 2008)

7 KÄYTTÄJÄKYSELY

Tekstiviestien käytöstä suoritettiin käyttäjäkysely opiskelijoiden kesken, mutta aikarajan sisällä opiskelijoiden vastauksia tuli suhteellisen vaatimattomasti. Kyselyitä tehtiin kaksi erilaista: yksi oppilaille ja toinen opettajille. Kyselyn täyttäminen ei vaatinut kokemusta tekstiviestien lähettämisestä tai vastaanottamisesta, vaan kyselyyn pystyivät kaikki vastaamaan. Liitteessä 1 näkyvät opiskelijoiden kysymykset ja vastaukset ja liitteessä 2 opettajien kysymykset ja vastaukset.

Opiskelijoiden kyselyssä ensimmäisenä kysymyksenä on: ”Oletko saanut tekstiviestillä informaatiota koululta esimerkiksi koskien kurssien muutoksia tai opettajien sairaspoissaoloja?”. Vastaajia oli 35 kappaletta, joista yksi oli saanut tekstiviestejä ja 34 ei ollut saanut. Kyselyn järjestämisen aikana palvelua oli käyttänyt useampikin opiskelija, mutta he eivät olleet syystä tai toisesta päässeet tekemään kyselyä. Tämän vuoksi ”Kyllä”-vastaajia oli melko rajoitetusti.

Toisena kysymyksenä on: ”Haluaisitko saada tietoa kursseista/opinnoista tekstiviestin muodossa?”. Tähän kysymykseen oli vastannut 34 opiskelijaa, sillä edellisessä kysymyksessä yksi oppilas oli käyttänyt jo kyseistä palvelua, jolloin hän ei saa tätä kysymystä kyselyssä. 24 opiskelijaa vastasi ”Kyllä” ja 10 vastasi ”En”. Tarvetta kyseiselle palvelulle siis selvästi on, ja ne opiskelijat jotka eivät halua käyttää kyseistä palvelua, voivat jättää laittamatta puhelinnumeronsa Moodlen tietoihin, jolloin he eivät tekstiviestejä saa.

Kolmantena kysymyksenä kysyttiin palvelun hyödyllisyydestä. Kysymyksenä on: ”Oletko kokenut palvelun hyödylliseksi, tarpeelliseksi tai onko se parantanut ajankäyttöäsi?”. Ainut opiskelija, joka oli käyttänyt jo kyseistä tekstiviestipalvelua, vastasi kaikkiin ”kyllä”, eli palvelu oli ollut hänelle tarpeellinen.

Opettajien kyselyn ensimmäisenä kysymyksenä on: ”Oletko lähettänyt tietoa tekstiviestillä (Repun tekstiviestilisäosa) opiskelijoille?”. Tähän kyselyyn vastasi yhteensä kuusi opettajaa, joista viisi oli käyttänyt palvelua.

Toisena kysymyksenä opettajille, jotka eivät olleet lähettänyt tekstiviestejä, on: ”Koetko tarpeelliseksi lähettää tietoa opiskelijoille tekstiviestin muodossa esim. opettajan äkkinäisestä sairauspoissaolosta tai kurssimuutoksesta?”. Ainut opettaja, joka ei ollut käyttänyt aikaisemmin tekstipalvelua, vastasi tähän kysymykseen ”kyllä”. Tämän jälkeen kysyttiin ”Miksi?” ja kommenttina oli: ”Tekstiviesti tavoittaa sähköpostia ja Repun tiedotteita nopeammin.”. Tämä on totta, sillä todennäköisempää on, että opiskelijalla on matkapuhelin mukana ja päällä kuin se, että hän tarkkailee sähköpostiaan tai Repun foorumia mahdollisen ilmoituksen takia.

Neljäntenä kysymyksenä oli: ”Onko lähettäminen ollut helppoa?”. Tähän kaikki viisi opettajaa vastasivat ”Kyllä”, joten tekstiviestipalvelun suunnittelu oli onnistunutta, koska tarkoituksena oli toteuttaa helppo ja yksinkertainen palvelu. Helppoudella tarkoitetaan tässä kohdassa sitä, että aikaisempaa tuntemusta Moodlen lisäosista tai massapostituspalveluista ei tarvita tekstiviestipalvelun sujuvan käytön onnistumiseksi.

Viimeisenä kohtana kyselyssä oli: ”Miten palvelua voisi parantaa?”. Opettajilta tuli erinäisiä kommentteja, kuten esimerkiksi ”kuvaviestit mukaan”. Tämä ei ole tällä hetkellä mahdollista, koska tuki on vain sms-viesteille. Seuraava ehdotus: ”Voisiko tekstiviestipalvelu olla vakiona Repun jokaisella sivulla. Nythän se piti ensin itse lisätä kurssille. Tämä on otettu huomioon, ja asiasta on keskusteltu Reppua ylläpitävän tahon kanssa mahdollisen toteuttamisen osalta.

Vastaus ”Puhelinnumeron kanssa on oltava erittäin tarkkana ettei mene väärin. Voisiko sitä kehittää vapaampaan suuntaan”, tarkoittaa lähettäjän nimen tai numeron syöttöä. Puhelinnumerot ovat absoluuttisia, ja niiden käyttö vaatii jossain määrin tarkkuutta. Tähän ei voida vaikuttaa. Mikäli lähettäjän numero

valittaisiin esimerkiksi listalta, toisi se tarpeetonta epäselkeyttä lähetyssivulle. Toivetta ei päätetty toteuttaa.

”Ääkkösten käyttäminen viesteissä” tarkoittaa sitä, että kirjoitettaessa viestiä, viestikentässä näkyy skandinaavisten merkkien tilalla epämääräisiä merkkejä. Kyseistä ongelmaa ei saatu tuotettua uudestaan käyttämällä eri selaimia ja eri merkistökoodauksia, joten ongelmaan ei päästy käsiksi.

Muut vastaukset olivat toteamuksia tai tilapäisiä katkoksia, kuten liitteessä 2 näkyy. Erään operaattorin kanssa oli lähettämisen kanssa suuria viiveitä, mutta tähän ei voitu millään vaikuttaa, sillä ongelma oli operaattorin päässä.

8 YHTEENVETO

Opinnäytetyön aihe oli minulle erittäin mieluinen, sillä itsekin opiskelijana olen saanut välillä kärsiä liian myöhään tulleista sairaspoissaoloilmoituksista ja näin ollen tullut koululle lukemaan opettajan lähettämän sähköpostin. Samaistuminen kohderyhmään toikin itselleni motivaatiota ja intoa kehittää lisäosaa käytettävyydeltään ja toiminnaltaan sopivaksi.

Lisäosan suunnittelu ja kehitys onnistuivat loistavasti. Ainoita ongelmia toi aikataulun venyminen, mikä johtui palveluun halutuista uusista ominaisuuksista. Yhtenä tärkeimmistä asioista, joita opin projektin aikana, pidän asiakkaan tarpeiden kartoittamista projektin alkuvaiheessa, jottei melkein valmiiseen tuotteeseen tarvitse loppumetreillä alkaa implementoimaan uutta toiminnallisuutta. Myös keskustelut ja palaverit Lahden ammattikorkeakoulun asianomaisten kanssa rohkaisivat kyseisen työn tekemisessä, sillä enemmistöllä tuntui olevan sellainen ajatus, että lisäosa on erittäin toivottu.

Projektin aikana opin erittäin paljon www-teknologioista ja erityisesti PHP- ja MySQL-kielestä. Tämä projekti toikin itseäni lähemmäksi web-ohjelmointia, sillä sitä käytiin melko vähän koulun kursseilla, minkä vuoksi oppiminen piti aloittaa melko pitkälti alusta alkaen.

Kehitysideoina näkisin älypuhelimiin ohjelmoitavan käyttöliittymän tekstiviestien lähetystä varten, sillä ne ovat alkaneet yleistyä räjähdysmäisesti. Isojen näyttöjen tuomat edut antaisivat tarpeeksi mahdollisuuksia juuri tämänkaltaisten sovellusten kehittämiseen.

LÄHTEET

Wikipedia. 2010.Moodle [viitattu 12.4.2010].

Saatavissa: http://en.wikipedia.org/wiki/Moodle

Moodle. 2010.About Moodle [viitattu 12.4.2010].

Saatavissa: http://docs.moodle.org/en/About_Moodle

Wikipedia. 2010.PHP [viitattu 21.10.2010].

Saatavissa: http://en.wikipedia.org/wiki/PHP

Moodle. 2010.Installing Moodle [viitattu 21.10.2010].

Saatavissa: http://docs.moodle.org/en/Installing_Moodle

Moodle. 2010.Development:Security [viitattu 18.11.2010].

Saatavissa: docs.moodle.org/en/Development:Security

Moodle. 2010. Development:Database schema introduction [viitattu 19.10.2010].

Saatavissa:

http://docs.moodle.org/en/Development:Database_schema_introduction

Moodle. 2010. Coding style [viitattu 20.10.2010].

Saatavissa: docs.moodle.org/en/Development:Coding_style

Yle. 2010. Hätätekstiviestit halutaan kaikkien koulujen käyttöön
[viitattu 5.11.2010]
Saatavissa:
http://yle.fi/alueet/helsinki/helsinki/2009/12/hatatekstiviestit_halutaan_kaikkien_koulujen_kayttoon_1266549.html

Tilastokeskus. 2008. [viitattu 2.11.2010]
Saatavissa: http://www.stat.fi/til/tvie/2008/tvie_2008_2009-06-09_tau_009_fi.html

Laurea-ammattikorkeakoulu. 2010. [viitattu 5.11.2010]
Saatavissa: http://markkinointi.laurea.fi/esitteet/Laurea_fakta_2010_2011.pdf

Nokia. 2001. MITA Mobile Internet Technical Architecture

Schiller J. 2001. Mobiilitietoliikenne

Appu A. 2002. Making use of PHP

Harold E. R. 2000. XML - Tehokäyttäjän opas. Suom. Saxberg P.

Ling R. 2004. The Mobile Connection The Cell Phone’s Impact on Society

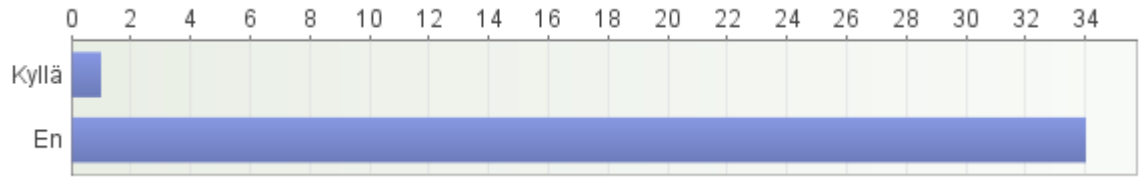
Heinisuo R. & Ranta I. 2007. PHP ja MYSQL - Tietokantapohjaiset verkkopalvelut

LIITTEET

Liite 1. Opiskelijoiden vastaukset

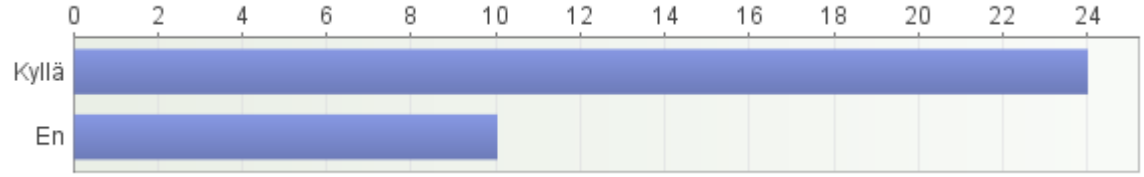
1. Oletko saanut tekstiviestillä informaatiota koululta esimerkiksi koskien kurssien muutoksia tai opettajien sairaspoissaoloja?

Vastaajien määrä: 35



2. Haluaisitko saada tietoa kursseista/opinnoista tekstiviestin muodossa?

Vastaajien määrä: 34



3. Oletko kokenut palvelun hyödylliseksi, tarpeelliseksi tai onko se parantanut ajankäyttöäsi? Voit valita useamman vaihtoehdon.

Vastaajien määrä: 1

4. Kommentit, parannusehdotukset

Vastaajien määrä: 1

- lol, rqrd

Liite 2. Opettajien vastaukset

1. Oletko lähettänyt tietoa tekstiviestillä (Repun tekstiviestilisäosa) opiskelijoille?
Vastaajien määrä: 6

2. Koetko tarpeelliseksi lähettää tietoa opiskelijoille tekstiviestin muodossa esim. opettajan äkkinäisestä sairauspoissaolosta tai kurssimuutoksesta?
Vastaajien määrä: 1

3. Miksi?
Vastaajien määrä: 1

- Tekstiviesti tavoittaa sähköpostia ja repun tiedotteita nopeammin.

4. Onko lähettäminen ollut helppoa?
Vastaajien määrä: 5

5. Miten palvelua voisi parantaa?
Vastaajien määrä: 5

- kuvaviestit mukaan
- Voisiko tekstiviestipalvelu olla vakiona Repun jokaisella sivulla. Nythän se piti ensin "itse" luoda.

- Puhelinnumeron kanssa on oltava erittäin tarkkana ettei mene väärin. Voisiko sitä kehittää vapaampaan suuntaan.
- Ääkkösten käyttäminen viesteissä.
- Aluksi oli katkoja palvelussa, ei enää.
- Hyvä palvelu, tiedottaminen on ehkä ollut vähäistä.

Liite 3. Lähetysformi