Ylläpitodokumentti

Aija

Helsinki 2.9.2005

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

Sisältö

1. Johdanto....3
2.Sanasto....3
3.Asennusohje....3
3.1 Paketin purkaminen....3
3.2 Tietokannan luonti ja tiedostojen käyttöoikeudet....4
3.3 Järjestelmän vakioiden määrittäminen....5
3.4 Session tiedostojen sijoitushakemiston asennus....5
3.6 Taustatoiminnon asentaminen....6
4. Käynnistysohje....7
5. Suunnitelman ylläpito....7
6. Koodin ylläpito....9
7. Testaus....11
8. Tietokanta....12

1. Johdanto

Tämä dokumentti käsittelee sellaisia asioita, joita mahdolliset Aija -projektin jatkoryhmät tai asiakkaan edustajat tarvitsevat muokatakseen ohjelmaa. Tätä dokumenttia ei ole tarkoitettu luettavaksi yksinään, vaan yhdessä suunnitteludokumentin ja ohjelmakoodin kanssa.

2.Sanasto

DTD	Document Type Definition, dokumentin tyyppimäärittely.
ISO International Standard	Kansainvälisen standardioimisjärjestön ISO:n julkaisema standardi.
Järjestäjä	Ryhmän muodostaja, esimerkiksi ohjaaja, esimies, tuutori.
Käyttäjä	Henkilö, joka on kirjautumassa sisään.
Osallistuja	Henkilö, joka kutsutaan ryhmään ja osallistuu tapaamisiin, esimerkiksi opiskelija
PHP	Eräs HTML-koodiin upotettava skriptikieli.
XML	Extensible Markup Language, eräs kuvauskieli.
XSL	Extensible Stylesheet Language, dokumentin tyylimäärittely.

3.Asennusohje

Järjestelmän ohjelmistoympäristö tulee olla suunniteludokumentissa luetellun mukainen (luku 2.3). Lisäksi www -ja tietokantapalvelimen tulee olla toiminnassa, millä tarkoitetaan että ne ovat valmiina palvelemaan asiakkaita.

3.1 Paketin purkaminen

Järjestelmä on pakattu “aija.tarz” pakettiin. Paketti tulee purkaa siihen hakemistoon johon järjestelmä tullaan asentamaan. Paketti puretaan komennolla

tar xvfz aija.tarz

Paketti purkautuu työhakemistoon johon ilmestyy hakemistot img/, inc/, log/, res/, sql/ sekä osajärjestelmien tiedostot.

3.2 Tietokannan luonti ja tiedostojen käyttöoikeudet.

Seuraavaksi luodaan tietokanta järjestelmää varten. Osoitteesta http://www.postgre.org/ löytyy tarkat ohjeet Postgre -tietokannan käytöstä ja sen asentamisesta. Lisäksi kaikille järjestelmän käyttäjille pitää antaa asianmukaiset oikeudet.

Paketista purkautui “install” skripti, jolla tietokannan luonti ja käyttöoikeuksien asetuksen voidaan suorittaa. Tutustu skriptin toimintaan ennen sen ajamista.

Skripti tulee suoritetaan komentoriviltä siinä hakemistossa mihin järjestelmä tullaan asentamaan. Skriptin suorittaminen edellyttää, että asentajalla on oikeudet “createdb” ja “chmod” -komentojen käyttöön. Tassa oletetaan, etta komennot löytyvat ilman, että niiden kutsuminen onnistuu ilman hakemistopolkujen antamista.

Skripti luo ensin tietokannan createdb komennolla. Oletusarvoisesti tietokannan nimeksi tulee "aija". Tässä vaiheessa asentajalta kysytään yleensä tietokannan käyttäjän salasanaa. Tässä dokumentissa viitataan jatkossa “aija” -nimiseen tietokantaan. Seuraavaksi skripti antaa asianmukaiset oikeudet kaikkiin tiedostoihin.

Skriptin suoritus edellyttää suoritusoikeuksia, joten ne tulee antaa komennolla

chmod u+x ./install

Skripti suoritetaan komennolla

./install

3.3 Järjestelmän vakioiden määrittäminen

Seuraavassa vaiheessa tarvitsemme seuraavat tiedot tietokantapalvelimesta:

- Käyttäjätunnus.
- Salasana.
- Palvelimen osoite.
- Palvelimen porttinumero.

Ohjelma käyttää globaaleja vakioita, jotka on tallennettu tiedostoihin. Tiedostot löytyvat “inc” -hakemistosta. Seuraavaksi asennamme vakiot.

Avaa “db.php” -tiedosto. Täydennä seuraavat muuttujat tähän tapaan:
define"DB_PASSWORD","salasana");

- DB_PASSWORD = käyttäjä salasana tietokantaan.
- DB_USERNAME = käyttäjätunnus tietokantaan.
- DB_SERVER = tietokantapalvelimen osoite. Yleensä "localhost", jos palvelin sijaitsee järjestelmän kanssa samassa ymparistössä.
- DB_NAME = tietokannan nimi, tässä "aija".
- DB_PORT = porttinumero jota tietokantapalvelin kuuntelee.

Muut vakiot ovat oletusarvoisesti oikein. älä muuta näitä. Muista tallentaa muutokset. Lopuksi avaa admin.php -tiedosto, joka sijaitsee asennuksen juurihakemistossa. Lisää aijan -ylläpitäjän salasana edellisten tapaan PASSWORD -vakion arvoksi.

3.4 Session tiedostojen sijoitushakemiston asennus.

Tarvitset hakemiston session muuttujien tallennusta varten. Tärkeintä on, etta aija -järjestelmällä on

käyttöoikeudet kyseiseen hakemistoon.

Inc -hakemistossa sijaitseva session.php komponentissa on lause “@session_save_path()”, johon tulee lisätä hakemistopolku johon session -muuttujat tallennetaan.
Esimerkiksi “@session_save_path("/home/tunnus/public_html/cgi-bin/tmp");

3.5 Tietokannan alustus.

Hakemistosta sql/ löytyy .sql -tiedostot jotka sisältävät skriptit tietokannan alustukseen. Voit ajaa skriptit “psql” ohjelmalla. Kirjaudu aija -tietokantaan komennolla "psql aija". Komennolla \i [tiedosto_nimi] voit suorittaa .sql -tiedostoja. Asetuksista riippuen voit joutua antamaan myös tiedoston hakemistopolun. Taulujen luonti voi antaa virheilmoituksia jotka johtuvat siitä että tietokanta on aluksi tyhjä. Tärkeintä on etta createTable.sql suoritetaan ensimmaisenä.

- createTable.sql luo taulut tietokantaan.crom
- locales.sql lisää tiedot locales tauluun.
- codes.sql lisää tiedot codes tauluun.
- values.sql lisää tiedot values tauluun.

Poistu ohjelmasta “\q” -komennolla.

3.6 Taustatoiminnon asentaminen.

Käyttöjärjestelmän tulee suorittaa backrun.php skripti päivittäin. Taman skriptin toiminta on kuvattu suunnitteludokumentissa. Skriptin suorituksesta voi huolehtia crontab. Voit muokata crontab -tiedostoasi komennolla

crontab -e

Mikäli asentajalla ei ole lupaa crontabin kayttöön täytyy asentajan kysya ylläpitäjältä oikeuksia tähän.

Voit määrätä kuinka usein taustatoiminto suoritetaan. Suosittelemme etta se suoritettaan päivittäin. Seuraavassa taustatoiminto asetetaan suoritettavaksi päivittäin klo 4:00.

Lisää crontab tiedostoosi seuraava rivi

0 4 * * * [php_tulkki] [backrun.php sijainti]

Esimerkiksi

0 4 * * * /usr/local/bin/php5 /home/aija/public_html/cgi-bin/backrun.php

Nyt olet asentanut aija järjestelmän.

4. Käynnistysohje

Järjestelmää ei tarvitse erikseen käynnistää. Ylläpitäjän tulee pitää huoli siitä, että tietokanta -ja www -palvelin ovat toiminnassa, ja että järjestelmää pystytään käyttämään web -selaimella. Ilman tietokantapalvelinta tai www- palvelinta on järjestelmä toimintakyvytön. Järjestelmän käyttö aloitetaan kutsumalla järjestelmän etusivua Web-selaimella. Esim http://palveluntarjoaja/~tunnus/cgi-bin/index.php.

5. Suunnitelman ylläpito

Järjestelmän JavaScript koodit eivät ole W3C:n laatiman standardin mukaisia. Tästä syystä järjestelmää ei voi käyttää kuin Mozilla FireFox selaimilla. Ongelma lienee siinä, että JavaScript käyttää standardista poistuneen funktion kutsua document.all(). Tällä kutsulla voidaan käsitellä ja manipuloida HTML-koodissa olevia elementtien attribuutteja ja sisältöä. Uudemman standardin mukaista kutsua getElementById() ryhmämme ei saanut toimimaan halutulla tavalla. Tämä standardista poikkeava vaihtoehto jouduttiin hyväksymään kompromissina, koska järjestelmän avaintoiminnot ovat täysin riippuvaisia JavaScriptistä, eikä ryhmällä ollut riittävästi pohjatietoa, eikä aikaa JavaScriptin

tarkkaan opetteluun. Kysymys, olisko toisenlaisen ohjelmistotekniikan tai ongelman lähestymistavan valinta parantanut tässä suhteessa järjestelmän käytettävyyttä, jää ylläpidon ratkaistavaksi.

Eräs määrittelyvaiheen vaatimuksista oli, että järjestelmä olisi sellainen mikä ei vaatisi suuria ylläpidollisia toimenpiteitä. Tämä asia mahdollistettiin luomalla komponentti (backrun.php), joka suorittaa vanhentuneiden tietojen poistamisen tietokannasta. Lisäksi luotiin käyttöliittymä (admin.php), jolla voidaan syöttää tietoja tietokantaan. Järjestelmästä puutuu toiminto, jolla tietoja voidaan poistaa tai päivittää olemassa olevia tietoja. Tällaiselle toiminnolle voi olla tarvetta mikäli halutaan esimerkisi vaihtaa ylläpitäjän sähköpostiosoitetta, tai määritellä uudestaan järjestäksi oikeuttavat sähköpostiosoitteet.

Järjestelmään voidaan siis lisätä tietoja admin.php:n avulla joka saa syötteekseen XML-dokumentteja. Nämä dokumentit ovat ennalta määriteltyjä ja admin.php osaa jäsentää vain
ja ainoastaan näitä dokumentteja. Dokumenttien rakenne ei kuitenkaan ole ihan ideaalinen suurien tieto määrien syöttöön, joten esimerkiksi uuden kielen lisääminen järjestelmään voi osoittautua varsin työlääksi. Tämä ylläpidon toiminnallisuus oli siis määriteltykin hyvin minimaaliseksi, joten siihen ei yritettykään saada optimaalista ratkaisua.

Järjestelmässä on tietoturvan kannalta muutamia epäkohtia, joihin tulisi tulevaisuudessa kiinnittää huomiota. Järjestelmään on mahdollista päästä sisään ilman henkilökohtaista salasanaa. Tämä johtuu eräästä asiakkaan vaatimuksesta. Osallistuja päästetään järjestemään sisään ilman salasanan vahvistuksia rekisteröitymisen yhteydessä. Tällä ominaisuudella on sellaisia vaikutuksia, että pahantahtoinen käyttäjä voi rekisteröityä sivullisen sähköpostiosoitteella ja päästä järjestelmään esittäytyen toisella identiteetillä. Tätä mahdollisuutta voi olla tulevaisuudessa syytä rajoittaa. Myöskään osajärjestelmien busines -logiikat eivät tarkista, että onko kutsuja todella järjestelmän käyttäjä ja onko hänellä oikeuksia tallentaa lähettämiään syötteitä tietokantaan. Tämän ongelman ratkaisu on kuitenkin melko helppo tehdä jälkikäteen ja se jääkin nyt ylläpidon tehtäväksi.

Järjestelmän käyttäjät jaetaan osallistujien ja järjestäjien ryhmiin sähköpostiosoitteen perusteella. Sähköpostiosoite perustainen jaottelu on kuitenkin varsin karkea jakoinen, joten mahdollisten järjestäjien lukumäärä voi olla vaikea rajata alle tuhanteen ja huolimattomalla valinnalla se voi kasvaa

miljooniin käyttäjiin.

6. Koodin ylläpito

Tietokantaan syöttämiseen vaadittavat XML -dokumentit löytyvät res -hakemistosta. XML- tiedostot ovat locales.xml, values.xml, codes.xml. Tiedoston nimi kertoo mitä tietokannan taulua muutos koskee.

Locales.xml noudattaa seuraavaa DTD:tä:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!ELEMENT texts (text+)>
<!ELEMENT text (name,value+)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ATTLIST value lang CDATA #REQUIRED>
```

Values.xml noudattaa seuraavaa DTD:tä:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!ELEMENT values (collection+)>
<!ELEMENT collection (name, value+,sorting)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT sorting EMPTY>
<!ATTLIST collection id CDATA #REQUIRED>
<!ATTLIST value lang CDATA #REQUIRED>
```

Codes.xml noudattaa seuraavaa DTD:tä:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!ELEMENT codes (collection+)>
<!ELEMENT collection (name, value)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ATTLIST collection id CDATA #REQUIRED>
```

Esimerkki codes.xml:

```
<?xml version='1.0' encoding="iso8859-1" '?>
<!DOCTYPE codes SYSTEM "codes.dtd">
<?xml-stylesheet type="text/xsl" href="codes.xsl"?>
<codes>
  <!-- Trusted emails -->
  <collection id="ADMINISTRATOR">
    <name>CS</name>
    <value>cs.helsinki.fi</value>
  </collection>
</codes>
```

Esimerkki locales.xml:

```
<?xml version='1.0' encoding="ISO-8859-1" standalone='yes'?>
<!DOCTYPE locales SYSTEM "locales.dtd">
```

```
<?xml-stylesheet type="text/xsl" href="locales.xsl"?>
<texts>
  <text>
    <name>ADMH</name>
    <value lang="fi">Valitse tiedostot jotka tallennetaan tietokantaan. Anna Salasana ja paina painiketta.
    </value>
    <value lang="en-US">Choose files which are inserted to database. Give password and press button.
    </value>
  </text>
</texts>
```

Esimerkki values.xml:

```
<?xml version='1.0' encoding="iso-8859-1" standalone='yes'?>
<!DOCTYPE values SYSTEM "values.dtd">
<?xml-stylesheet type="text/xsl" href="values.xsl"?>
<values>
  <!-- Proprieties -->
  <collection id="PREFS">
    <name>0</name>
    <value lang="fi">ei tiedossa</value>
    <value lang="en-US">unknown</value>
    <sorting/>
  </collection>
</values>
```

XML-dokumenteilla on omat XSL -tyylitiedostonsa. Nämä voitaisiin yhdistää myös yhdeksi tiedostoksi.

Locales.xsl:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              version="1.0">
  <xsl:output method="html"/>
  <xsl:template match="texts">
    <html>
      <body bgcolor="#FFFFFF">
        <h3><a name="queries">SQL -queries</a></h3>
        <xsl:apply-templates select="text">
          <xsl:sort select="name" data-type="text" order="ascending"/>
        </xsl:apply-templates>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="text">
    <xsl:for-each select="value">
      INSERT INTO Locales VALUES ('<xsl:apply-templates select="../name"/>',
      '<xsl:apply-templates select="@lang"/>','<xsl:apply-templates/>');<br/>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

Codes.xsl:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              version="1.0">
  <xsl:output method="html"/>
  <xsl:template match="codes">
    <html>
      <body bgcolor="#FFFFFF">
        <h3><a name="queries">SQL -queries</a></h3>
```

```
      <xsl:apply-templates select="collection">
        <xsl:sort select="@id" data-type="text" order="ascending"/>
      </xsl:apply-templates>
     </body>
    </html>
  </xsl:template>
  <xsl:template match="collection">
    <xsl:for-each select="value">
      INSERT INTO Codes VALUES ('<xsl:apply-templates select="../@id"/>',
      '<xsl:apply-templates select="../name"/>','<xsl:apply-templates/>',
       <xsl:choose>
         <xsl:when test="../sorting[text()!= "]">
           <xsl:apply-templates select="../sorting"/>
         </xsl:when>
         <xsl:otherwise>
           NULL
         </xsl:otherwise>
       </xsl:choose>);<br/>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

Values.xsl:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
              version="1.0">
  <xsl:output method="html"/>
  <xsl:template match="values">
    <html>
      <body bgcolor="#FFFFFF">
       <h3><a name="queries">SQL -queries</a></h3>
       <xsl:apply-templates select="collection">
         <xsl:sort select="@id" data-type="text" order="ascending"/>
       </xsl:apply-templates>
      </body>
    </html>
  </xsl:template>
  <xsl:template match="collection">
    <xsl:for-each select="value">
      INSERT INTO Values VALUES ('<xsl:apply-templates select="../@id"/>',
      '<xsl:apply-templates select="../name"/>','<xsl:apply-templates select="@lang"/>',
      '<xsl:apply-templates/>',
       <xsl:choose>
         <xsl:when test="../sorting[text()!= "]">
           <xsl:apply-templates select="../sorting"/>
         </xsl:when>
         <xsl:otherwise>
           NULL
         </xsl:otherwise>
       </xsl:choose>
       );<br/>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

7. Testaus

Testaus on kerrottu yksityiskohtaisemmin testausdokumentissa. Järjestelmän yksikkötestaus suoritettiin osaksi eri ympäristössä kuin mihin se sijoitettiin. Tästä seurasi odottamattomia ongelmia, mikä pidensi integrointitestausta. Testauksen toistettavuutta olisi parantanut se, että testaus olisi automatisoitu niin

pitkälle kuin mahdollista. Tähän ryhmällä oli kuitenkin rajalliset resurssit, ja vain muutamalle komponentille tehtiin testausskripti. Testauksen automatisoinnilla oltaisiin voitu parantaa virheiden löytymisen mahdollisuutta, kun komponenttien olisi pitänyt suoriutua testaajan antamista ehdoista, tarkalleen saman testin mukaan kerrasta toiseen. Järjestelmälle luodut testausskriptit ovat emaiTest.php ja dbTest.php. Skripti ajetaan web -selaimella kutsumalal sitä. Skripti tulostaa ruudulle virheilmoituksen virheen sattuessa.

Järjestelmän tärkeimpien toimintojen testaus osoittautui kuitenkin haasteelliseksi, koska toimintojen tulos on riippuvainen käyttäjän antamista syötteistä. Syötteet annetaan käyttöliitymän kautta, mikä lisää erilaisten mahdollisten tapahtumajonojen kombinaatioiden määrää. Tästä syystä ohjelmaan jää todennäköisesti virheitä, jotka aiheutuvat sellaisten käyttäjän valintojen kombinaatioista, mitä ei testauksessa ehditty huomioimaan.

Taustatoiminnon testaus jäi vaillinaiseksi, mikä olisi syytä tarkistaa järjestelmän käyttöönoton jälkeen. Komponentti tekee sen mitä on määriteltykin, mutta mahdollisia järjestemän ympäristöstä aiheutuvia ongelmia ei ole testattu. Mitä tapahtuu kun ohjelmistoalusta kaatuu komponentin suorituksen kesken?

Lisä selvitystä vaatii myös se, että miten järjestelmä käyttäytyy kun sen osajärjestelmiä tai business -logiikoita kutsutaan virheellisillä parametreilla. Toinen huomion arvoinen seikka on myös, että aiheuttaako käyttäjän koneelle lähetettyjen html-sivujen syötekenttien manuaalinen muokkaminen tietoturvaukkoja, tai osaako taustatoiminto poistaa tällä tavalla syötetyt tiedot tietokannasta.

8. Tietokanta

Tietokannan ylläpito voi osoittautua ylläpitäjälle työlääksi. Tietokantaan voi syöttää tietoja XML -dokumenttien avulla, mutta tietojen päivitys täytyy tehdä manuaalisesti. Mikäli tietokantaan tehtävät muutokset voivat vaikuttaa myös koodiin. Taulujen nimien vaihtamisen tai taulujen lisäämisen jälkeen on nimet syytä muuttaa tai lisätä myös tietokantakomponentin vakioiden joukkoon.