Ylläpitodokumentti

Kohahdus

Helsinki 14.12.2006
Ohjelmistotuotantoprojekti
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Taro Morimoto, Projektipäällikkö
Tuomas Palmanto, Vaatimusmäärittelyvastaava
Mikko Kinnunen, Suunnitteluvastaava
Markus Kivilä, Koodivastaava
Jari Inkinen, Testausvastaava
Paula Kuosmanen, Dokumenttivastaava

Asiakas

Teemu Kerola

Johtoryhmä

Sanna Keskioja

Kotisivu

http://www.cs.helsinki.fi/group/kohahdus

Versiohistoria

Versio	Päiväys	Tehdyt muutokset
0.1	11.12.2006	Ensimmäinen versio
0.2	13.12.2006	Viimeinen draft versio
1.0	14.12.2006	Valmis versio

Sisältö

1. Johdanto

Kohahdus on järjestelmä automaattisesti tarkastettavien TTK-91-konekielen harjoitustehtävien luomiseen ja ratkaisemiseen. Järjestelmä on tarkoitettu käytettäväksi opetuksen tukena, opetettaessa Tietokoneen toiminta -kurssia. Tietojenkäsittelytieteen opettajat voivat tehdä järjestelmään uusia tehtäviä ja määritellä kuinka ne tarkastetaan automaattisesti. Tietokoneen toiminta -kurssin opiskelijat ja kurssin tehtävistä kiinnostuneet opiskelijat ja itseopiskelijat voivat ratkaista tehtäviä ja saada palautetta niiden onnistumisesta. Lisäksi opiskelijoiden tekemät tehtävät tilastoidaan.

Järjestelmä on toteutettu Java/JSP/JSTL/JavaScript ohjelmointikielillä ja sitä ajetaan Tomcat -palvelimessa. Tomcatin lisäksi erillisiä prosesseja ei tarvita. Järjestelmä tietokantana toimii laitoksen Oracle 10.

Tämän dokumentin tarkoituksena on kuvata lyhyesti ylläpitoon liittyviä asioita. Järjestelmän tarkempi kuvaus löytyy suunnitteludokumentista, joka on päivitetty projektin loppuvaiheessa vastaamaan lopullista tuotosta.

Opiskelijan käyttäliittymä löytyy osoitteesta:

http://sysdb.cs.helsinki.fi:10025/titotrainer/www/login.jsp

Opettajan käyttäliittymä löytyy osoitteesta:

http://sysdb.cs.helsinki.fi:10025/titotrainer/www/login.jsp?role=teacher

2. Sanasto

TTK91=Auvo Häkkisen kehittämä ohjelmointikieli, joka läheisesti muistuttaa symbolista konekieltä.

Järjestelmä=Projektimme tuotos, Kohahdus

TitoTrainer=Kohahduksen tuotoksen nimi

eAssari=Tietokantapohjainen ympäristö ohjelmallisesti tarkastettavien harjoitus- ja koetehtävien suorittamiseen

Titokone=Koski-nimisen Ohjelmistotuotantoprojektiryhmän vuonna 2004 rakentama järjestelmä konekielisten ohjelmien kääntämiseen ja suorittamiseen.

Kriteeri=Sääntö jonka mukaan tehtävän oikeellisuus tarkistetaan. Kriteereitä voi olla monta yhdelle tehtävälle.

Aihepiiri=Tehtävälle täytyy määritellä aihepiiri, johon tehtävä kuuluu.

HttpUnit=Testaustyökalu www-sivuille (kuten servletit). Perustuu JUnit:iin.

JSPUnit=Työkalu erityisesti JSP-sivujen testaamiseen. Perustuu JUnit:iin.

JSP=Java Server Pages. Dynaamisten sivujen tekoon kehitetty ohjelmointikieli.

3. Asennus ja asetusten määrittäminen

3.1 Ympäristö

Tuontantoympäristö sijaitsee sysdb.cs.helsinki.fi -palvelimella. Tomcat -palvelin sijaitsee /home/tkt_koha -hakemistossa. Ympäristössä käytetään Javan versoita 1.5 ja Tomcatin versiota 5.5.7. Huom! JSP-sivujen kääntäminen Tomcatissä tapahtuu Javan versiolla 1.4.

Testiympäristö sijaitsee db.cs.helsinki.fi -palvelimella. Tomcat -palvelin sijaitsee /home/tkt_koha -hakemistossa. Ympäristössä käytetään Javan versoita 1.5 ja Tomcatin versiota 4.1.18. Huom! JSP-sivujen kääntäminen Tomcatissä tapahtuu Javan versiolla 1.4.

Molemmissa ympäristöissä on käytetään samaa tietokantaa eli testiympärisön omaa tietokantaa ei ole olemassa.

3.2 Tietokanta

- Tietokantana on Oracle 10, joka löytyy osoitteesta **`bodbacka.cs.helsinki.fi:test`** käyttäjätunnus ja salasana ovat **`kohahdus/`*`SALASANA`***
- Salasana löytyy järjestelmän asennus CD:ltä.
- Titokannan luontiin käytetyt SQL-scriptit löytyvät tiedostosta eassari.sql
- Jos kanta on tyhjänä, niin ensin täytyy luoda oletustehtävät, jotta voitaisiin luoda uusia tehtäviä.
    1. Siirry hakemistoon: .../tomcat/webapps/titotrainer/WEB-INF/classes/
    2. Luo oletustehtävät ajamalla komento:

    java fi.helsinki.cs.kohahdus.criteria.TaskMaker jdbc:oracle:thin:kohahdus/PASS@bodbacka.cs.helsinki.fi:1521:test kohahdus PASS
- Lisäksi ensimmäinen käyttäjä luodaan kannan luonnin yhteydessä. Käyttäjän käyttäjätunnus on admin ja salasana on admin. Salasana täytyy muuttaa heti. (Huom. Käytetty hetu on generoitu)

3.3 Asennus Tomcat -palvelimelle

Järjestelmä voidaan asentaa seuraavia ohjeita noudattaen.

1. Asennetaan Java 1.5.
2. Asennetaan Tomcat 4.x -palvelin.
3. Luodaan palvelimelle uusi TitoTrainer -konteksti:
    1. Puretaan titotrainer.jar paketti hakemistoon .../tomcat/webapps/
    2. Lisätään .../tomcat/conf/server.xml tiedostoon seuraava rivi (uusi konteksti):

        <Context path="/titotrainer" docBase="titotrainer" debug="0" reloadable="false" />
4. Tarkistetaan .../tomcat/webapps/titotrainer/WEB-INF/web.xml tiedostosta, että järjestelmän asetukset ovat kohdallaan. (Salasana löytyy järjestelmän asennus CD:ltä)Alla on listaus oletusasetuksista:

```
<filter>
```

```
        <filter-name>TitoInitializer</filter-name>
        <filter-class>fi.helsinki.cs.kohahdus.TitoInitializer</filter-class>
                <init-param>
                        <param-name>context-name</param-name>
                        <param-value>titotrainer</param-value>
                </init-param>
                <init-param>
                        <param-name>context-path</param-name>
                        <param-value>/home/tkt_koha/tomcat/webapps/titotrainer/</param-value>
                </init-param>
                <init-param>
                        <param-name>language-properties</param-name>
                        <param-value>WEB-INF/xml/properties.xml</param-value>
                </init-param>
                <init-param>
                        <param-name>db-string</param-name>
                        <param-
value>jdbc:oracle:thin:kohahdus/SALASANA@bodbacka.cs.helsinki.fi:1521:test</param-value>
                </init-param>
                <init-param>
                        <param-name>db-username</param-name>
                        <param-value>kohahdus</param-value>
                </init-param>
                <init-param>
                        <param-name>db-password</param-name>
                        <param-value>SALASANA</param-value>
                </init-param>
                <init-param>
                        <param-name>smtp-server</param-name>
                        <param-value>localhost</param-value>
                </init-param>
                <init-param>
                        <param-name>smtp-port</param-name>
                        <param-value>25</param-value>
                </init-param>
    </filter>
```

4. Ylläpito

4.1 Lokin kirjoitus

Järjestelmä kirjoittaa lokia tiedostoon: .../tomcat/logs/catalina.out. Tähän tiedostoon kirjautuu tärkeimmät tapahtumat ja tarkemmat virheilmoitukset.

4.2 Vikatilanteet

Vikatilanteiden sattuessa tarkista ensin järjestelmän loki ja yritä selvittää, että mistä vika johtuu. Jos vika ei selviä, niin Tomcat -palvelin kannattaa käynnistää uudelleen. Uudelleenkäynnistys tapahtuu ajamalla scriptit:

1. stop-tomcat
2. start-tomcat

4.3 Monikielisyys ja webbisivujen tekstien muuttaminen

Järjestelmän monikielisyyden toteutus on kuvattu suunnitteludokumentissa luvussa 6 Monikielisyys. Jos JSP-sivujen tekstejä halutaan muuttaa, niin tulee ensin etsiä ks. JSP-sivulta seuraavanlainen rivi:

```
ResourceBundle menu = LanguageManager.getTextResource(lang , "sivun_avain");
```

.../tomcat/webapps/titotrainer/WEB-INF/xml/properties.xml tiedostosta löytyy “sivun_avaimella” tiedosto, jossa on listattuna xml-muodossa ks. sivun tekstit.

Tekstien muuttamisen jälkeen tulee Tomcat -palvelin uudelleenkäynnistää ajamalla scriptit:

1. stop-tomcat
2. start-tomcat

5. Järjestelmän jatkokehitys

5.1 Versionhallinta ja lähdekoodit

Järjestelmän lähdekoodit ja resurssit sijaitsevat CVS:ssä osoitteessa:

```
:extssh:your_username@melkki.cs.helsinki.fi:/group/home/kohahdus/cvsroot
```

Jos CVS:ää halutaan käyttää jatkokehityksessä, niin täytyy käyttäjä lisätä `kohahdus` -käyttäjäryhmään.

5.2 Ideoita jatkokehitykseen

Tehvävien monikielisyyden parantaminen

Tehtävien monikielisyys on tällä hetkellä toteutettu siten, että jokaisesta tehtävästä on olemassa vain yksi versio. Esimerkiksi englanninkieliset ja suomenkieliset tehtävät ovat kokonaan erillisiä tehtäviä. Tulevaisuudessa voisi olla hyvä vaihtoehto kehittää “aidosti” monikielisiä tehtäviä siten, että jokaiseen tehtävään “sisältyy” monta kieliversiota.

Monikielisyyden tekstit tietokantaan

Tällä hetkellä monikielisyyden toteuttamat tekstit ovat tiedostoissa. Tämä siksi, koska tekstien muuttamiselle ei ole olemassa erillistä käyttöliittymää.

Tulevaisuudessa tekstit voitaisiin laittaa tietokantaan ja rakentaa käyttöliittymä tekstien muokkaamiselle.

Itse TitoTrainerin monikielisyyden käyttöä ei tarvitsisi muuttaa. Sen sijaan tekstien lataaminen järjestelmän alustuksen yhteydessä toteutettaisiin toisella tavalla. Eli metodi fi.helsinki.cs.kohahdus.languages.LanguageManager.loadTextResourses() korvattaisiin metodilla, joka hakee tekstit tietokannasta tekstitiedostojen sijasta.

Dynaaminen tehtävätyyppi

Asiakkaan jatkokehitystoiveissa mainitaan dynaaminen tehtävätyyppi. Dynaaminen tehtävätyyppi ei sinänsä ole uusi tehtävätyyppi kuten Ohjelmointitehtävä ja Fill-In tehtävä, vaan ainoastaan laajennus nykyisiin tyyppeihin. Dynaamisessa tehtävässä ohjelmalle arvotaan satunnainen näppäimistösyöte, kun nyt ohjelmalle annetaan kiinteästi määritellyt julkiset ja salaiset syötteet.

Dynaamisen syötteen voisi toteuttaa answer_task.jsp sivulla siten, että aina kun tehtävä ladataan näytettäväksi, arvotaan näppäimistösyöte uusiksi. Tehtävänluontisivua ja ScreenOutputCriterion-luokkaa täytyy myös muuttaa, jotta ne hyväksyvät numeroiden lisäksi myös satunnaisen syötteen paramerit (esim määritellään syöte ”1, 5, RAND, RAND(1,100) , RAND, RAND(-5,5), 0”).

Mikäli TitoTrainer joskus kirjoitetaan kokonaan uusiksi (esim. valmiiseen Moodle tai Eassari kehykseen), on syytä harkita erillisten salaisen ja julkisen syötteen korvaamista yhdellä satunnaisuuteen kykenevällä syötteellä. Tämä yksinkertaistaisi tehtävien tarkistuskriteerejä ja itse tarkistusprosessia, sekä olisi mahdollisesti helpompi opiskelijoiden ymmärtää.