Testaussuunnitelma

Karstula

Helsinki 20.4.2007

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Juha-Pekka Juutilainen
Hannu Kukko
Antto Mäkinen
Antti Rajasärkkä
Ari Raunio
Mika Tantarimäki

Asiakas

Eija Mäntyharju

Johtoryhmä

Juha Taina
Kimmo Simola

Kotisivu

`http://www.cs.helsinki.fi/group/karstula`

Versiohistoria

Versio	Päiväys	Tehdyt muutokset
0.1	26.2.2007	Ensimmäinen versio
0.2	5.3.2007	Kirjoitettu kattavammin
0.3	14.3.2007	Korjailtu puutteita

Sisältö

1 Johdanto

Tämä on Karstula-ohjelmistotuotantoprojektin testaussuunnitelma. Projektin tarkoituksena on kehittää Karstulan evankeliselle kansanopistolle ohjelma, jolla voidaan harjoitella lääkelaskuja ja suorittaa lääkelaskutestejä.

Testaus tehdään kolmessa eri vaiheessa: yksikkötestaus, integrointitestaus ja järjestelmätestaus. Yksikkötestaus keskittyy luokkien sisäiseen toimintaan. Integrointitestaus keskittyy luokkien ja komponenttien rajapintoihin. Järjestelmätestauksessa järjestelmää testataan kokonaisuutena.

2 Sanasto

Cobertura	Testaustyökalu kattavuustestaukseen
EUCT	Extended Use Case Test, laajennettu käyttötapaus
JUnit	Testaustyökalu yksikkötestaukseen

3 Resurssit

Testaus toteutetaan JUnit-työkalun avulla. Kattavuustestaus tehdään Cobertura-työkalulla. Testitapaukset tehdään lähdekoodihakemiston kanssa rakenteeltaan samanlaiseen, mutta rinnakkaiseen hakemistoon. Tiedoston `src/karstula/gui/Main.java` testaus tulisi siten tiedostoon `test/karstula/gui/TestMain.java`.

Testit suoritetaan juurihakemistossa komennolla `ant test`, joka kääntää ohjelmaan Coberturan kattavuusinstrumentoinnin ja ajaa JUnit testit. JUnit-testiraportti tulee tiedostoon `test/reports/junit-html/index.html` ja Coberturan kattavuusraportti tiedostoon `test/reports/coverage-html/index.html`. Niitä voi selata internetselaimella.

Eri testausvaiheiden hyväksymiskriteerit testataan laitoksen koneilla sekä Windows- että Linux-ympäristössä.

Yksikkötestit kirjoittaa koodin tekijä. Toteutuksessa käytetään oletettavasti samaa työnjakoa kuin oli suunnitteluvaiheessa, joten yksikkötestaus jaetaan seuraavasti: Antti+Hannu käyttöliittymä, Ari+Juha-Pekka logiikka, Antto+Mika tietokanta.

Integrointitestaus tehdään kahdessa osassa. Käyttöliittymän ja logiikan integroinnin testaavat Antti+Hannu+Ari, logiikan ja tietokannan integroinnin testaavat Juha-Pekka+Antto+Mika.

Järjestelmätestaus jää kaikkien tehtäväksi.

```
package karstula.gui;

import junit.framework.*;

public class TestMain extends TestCase {
    public void testSomething() {
        Main.myMethod();
    }

    public void testElse() {
        assertTrue( false );
    }
}
```

Kuva 1: Esimerkki JUnit-yksikkötestistä

4 Yksikkötestaus

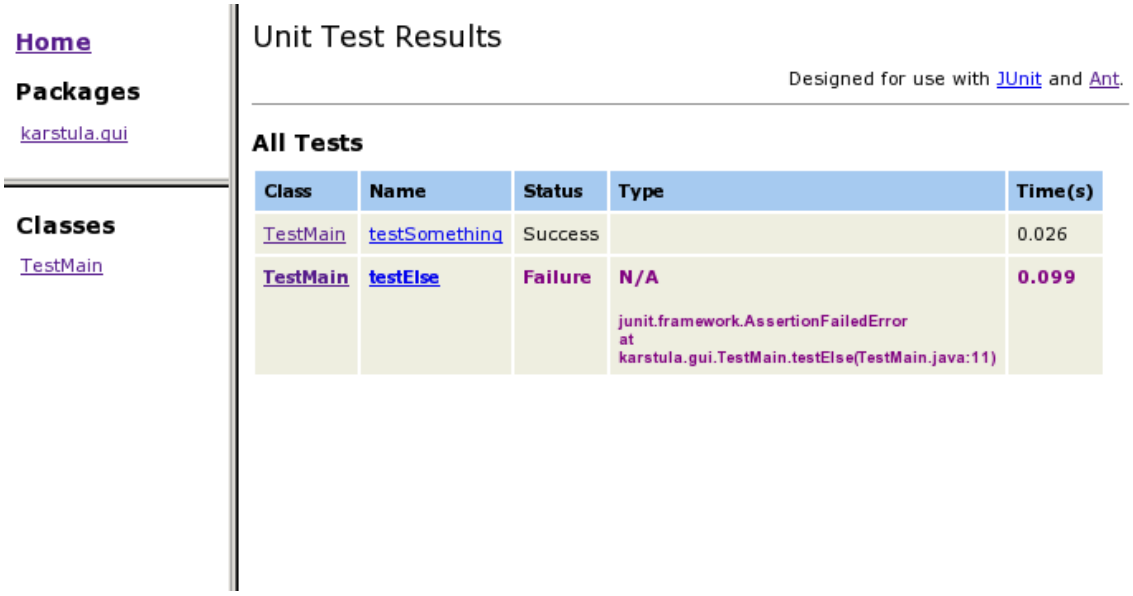
Yksikkötestauksessa testataan järjestelmän pienimmät osat, eli luokat. Yksikkötestauksen tarkoitus on varmistaa, että luokat toimivat ja käyttäytyvät toivotulla tavalla.

Kuvassa 1 on esimerkki yksikkötestistä. Tiedoston ja luokan nimeen tulee etuliite Test. Testiluokat perivät JUnit-paketin luokasta TestCase, joka tarjoaa useita eri assert-metodeja. Testit kirjoitetaan test-alkuisiin metodeihin. JUnit käyttää Javan reflection-tekniikkaa (JVM palauttaa tietoa olioista ja niiden metodeista) etsimään test-alkuisia metodeja ja suorittaa ne.

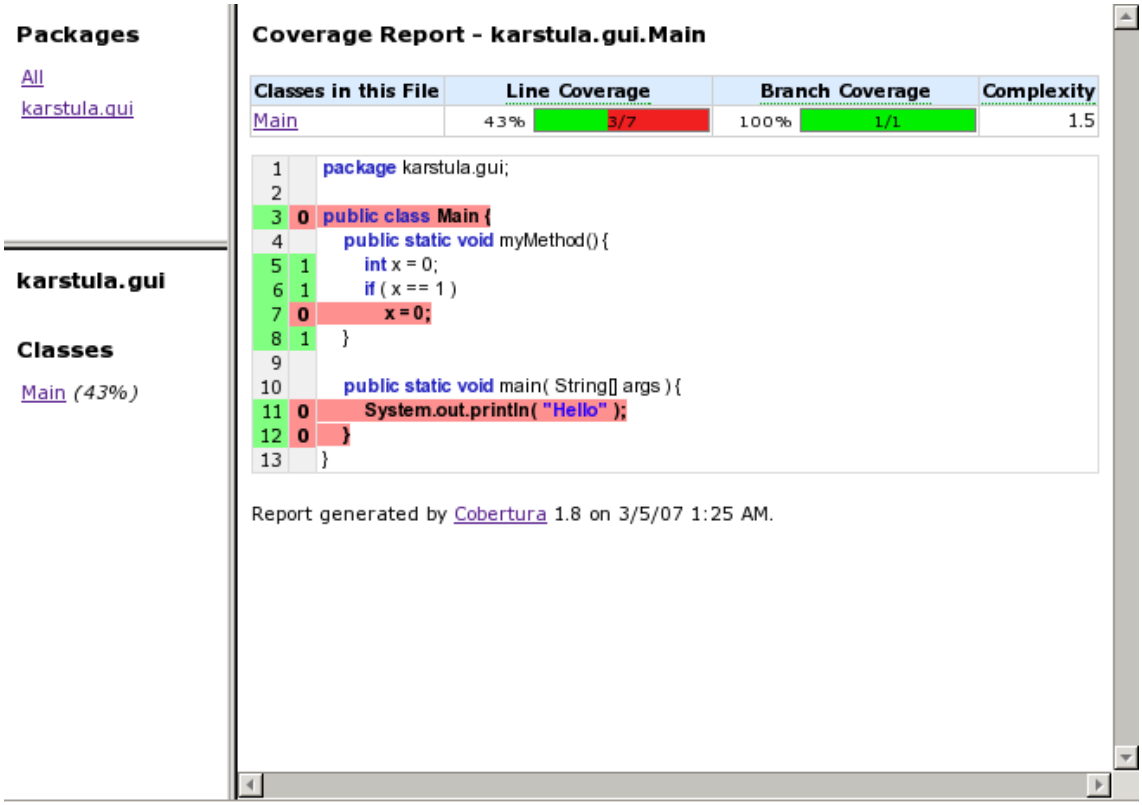
Kuvassa 2 on esimerkki JUnit-työkalun tulosteesta. JUnit käsittelee kahdenlaisia virheitä. *Failure* tarkoittaa tilannetta, jossa jokin assert-metodi on heittänyt poikkeuksen. *Error* tarkoittaa tilannetta, jossa poikkeus on heitetty jostain muualta.

Kattavuustestaus tehdään Cobertura-työkalulla. Cobertura mittaa kattavuuden lisäämällä laskureita sopiviin kohtiin ohjelmakoodissa. Cobertura ei vaadi ohjelman uudelleenkääntämistä, vaan instrumentointi lisätään suoraan tavukoodiin. Kuvassa 3 on esimerkki Coberturan tuottamasta kattavuusraportista. Vasemmanpuoleinen numerosarake koostuu rivinumeroista, jotka on merkattu vihreällä jos rivi on instrumentoitu. Oikeanpuoleinen sarake näytää rivin suorituskerrat. Rivi on punainen jos sitä ei ole suoritettu kertaakaan, muuten vihreä.

Lausekattavuus (line coverage) tarkoittaa suoritettujen lauseiden määrää suhteessa instrumentoituihin lauseisiin. Coberturan haaraumakattavuus (branch coverage) tarkoittaa ilmeisesti vain ehtoon saapumisten lukumäärää, dokumentaatio ei täsmennä tätä.



Kuva 2: Esimerkki JUnit-työkalun testaustuloksesta



Kuva 3: Esimerkki Coberturan kattavuusraportista

4.1 Lähestymistapa

Luokan metodit testataan tyypillisillä sallituilla syötteillä. Metodeista testataan myös ekvivalenssiluokat, eli parametrien eri loogiset arvojoukot. Parametrien arvot rajojen ympärillä ja `null`-arvot testataan. Lisäksi metodien käyttäytyminen virheellisillä syötteillä testataan.

4.2 Testattavat kohdat

Luokkien kaikki julkiset metodit testataan. Triviaalit metodit, joissa mikään ei voi mennä vikaan, ei tarvitse erikseen testata. Tällaisia ovat esimerkiksi set- ja get- metodit tai muut toiminnaltaan ilmeiset metodit.

4.3 Hyväksymiskriteerit

Yksikkötestaus on riittävä, kun luokkien kaikki toiminnot on testattu, määritellyt poikkeukset on testattu ja lausekattavuus ylittää 90%. Käyttöliittymää ei ehkä ole mahdollista testata näin kattavasti yksikkötestein.

5 Integrointitestaus

Integrointitestauksessa testataan osajärjestelmien väliset rajapinnat. Tarkoitus on testata osajärjestelmien yhteistoiminta ja varmistaa, että rajapinnat on toteutettu oikein ja että niitä käytetään oikein.

5.1 Lähestymistapa

Integrointitestaus tehdään alhaalta ylös. Kun kaksi osajärjestelmää on hyväksytysti yksikkötestattu, niiden välillä voidaan suorittaa integrointitestaus. Testitapaukset luodaan suunnitteludokumentin rajapintakuvausten perusteella ja kirjoitetaan samaan tapaan kuin yksikkötestit. Ensin testataan Tietokannan ja Logiikan integrointi. Sitten testataan Logiikan ja Käyttöliittymän integrointi.

5.2 Testattavat kohdat

Kaikki osajärjestelmien väliset rajapinnat testataan. Jokaiselta osajärjestelmältä selvitetään tarjotut ja vaaditut rajapinnat, jotka sitten testataan kuten metodit yksikkötesteissä. Rajapintojen parametrien arvoalueet selvitetään ja testataan ne.

5.3 Hyväksymiskriteerit

Integrointitestaus on riittävä, kun kaikki osajärjestelmien rajapinnat on testattu.

6 Järjestelmätestaus

Järjestelmätestauksessa järjestelmää testataan kokonaisuutena. Tarkoitus on testata järjestelmän toiminta sekä vaatimusdokumentin vaatimusten täyttyminen.

6.1 Lähestymistapa

Järjestelmä testataan käyttöliittymän kautta.

6.2 Testattavat kohdat

Järjestelmä testataan vaatimusdokumentin käyttötapauksia vastaavilla laajennetuilla käyttötapauksilla (Extended Use Case Test). Laajennetun käyttötapauksen tarkoitus on muuttaa tavallinen käyttötapaus sellaiseen muotoon, joka voidaan helposti testata.

6.3 Laajennetut käyttötapaukset

Tässä luvussa luetellaan vaatimusdokumenttia vastaavat laajennetut käyttötapaukset. Laajennettujen käyttötapausten tarkoitus on muokata vaatimusdokumentin käyttötapaukset sellaiseen muotoon, jossa ne on mahdollista testata järjestelmällisesti. Niissä määritellään muuttujien arvoalueet, syötteiden ja tulosten suhteet ja muut vastaavat oleelliset parametrit.

6.3.1 Teorian harjoittelu

Sidosryhmä: Ohjelman käyttäjä.

Kuvaus: Käyttäjä pääsee lukemaan teoriadokumentteja painamalla jonkin laskutyypin Teoria-painiketta.

Alkutila: Ohjelma on päävalikossa.

Skenaariot:

Syöte: Käyttäjä painaa yhtä Teoria-painikkeista.
Tulos: Laskutyypin teoriamateriaali näkyy kuvaruudulla.

6.3.2 Laskujen harjoittelu

Sidosryhmä: Ohjelman käyttäjä.

Kuvaus: Käyttäjä harjoittelee laskuja kirjoittamalla vastauksen vastauskenttään ja painamalla Vastaa-painiketta.

Alkutila: Ohjelma on harjoitteluosiossa.

Skenaariot:

Syöte: Oikein muotoiltu vastaus, oikea vastaus.
Tulos: Kuvaruudulla näkyy seuraava kysymys.

Syöte: Oikein muotoiltu vastaus, väärä vastaus.
Tulos: Kuvaruudulle tulee uusi vihje, yhteensä enintään 2, jolloin molemmat vihjeet näkyvät.

Syöte: Väärin muotoiltu vastaus.
Tulos: Kuvaruudulla näkyy virheilmoitus vastauksen väärästä muodosta.

Syöte: Käyttäjä pyytää vihjeen.
Tulos: Kuvaruudulle tulee uusi vihje, yhteensä enintään 2, jolloin molemmat vihjeet näkyvät.

Syöte: Käyttäjä ohittaa kysymyksen kolmannen väärän vastauksen jälkeen.
Tulos: Kuvaruudulle tulee uusi kysymys.

Syöte: Käyttäjä painaa "Näytä vastaus-painiketta kolmannen väärän vastauksen jälkeen.
Tulos: Kuvaruudulle tulee kysymyksen oikea vastaus ja laskukaava.

Syöte: Käyttäjä avaa teoriaosion.
Tulos: Kuvaruudulle tulee teoriaosio. Kysymys katsotaan väärin vastatuksi.

Syöte: Käyttäjä vastaa oikein helppoon kysymykseen.
Tulos: Kuvaruudulle tulee normaali kysymys.

Syöte: Käyttäjä vastaa väärin tai ohittaa helpon kysymyksen.
Tulos: Kuvaruudulle tulee uusi helppo kysymys.

Syöte: Käyttäjä on valinnut tietyn laskutyypin harjoituksen.
Tulos: Kuvaruudulle tulevat kysymykset ovat valittua laskutyyppiä.

Syöte: -
Tulos: Kysymyksessä näkyy vastauksen vaatima yksikkö.

6.3.3 Harjoitteluosion yhteenveto

Sidosryhmä: Ohjelman käyttäjä.

Kuvaus: Käyttäjä saa yhteenvedon tuloksista ja suosituksen harjoituksen päätyttyä.

Alkutila: Ohjelma on harjoitteluosiossa.

Skenaariot:

Syöte: Käyttäjä lopettaa harjoittelun.
Tulos: Kuvaruudulla näkyy yhteenveto tuloksista ja suositus.

Syöte: Käyttäjä harjoittelee kaksi kertaa.
Tulos: Yhteenvedon tulokset ovat viimeisen harjoittelun tuloksia.

6.3.4 Testin tekeminen

Sidosryhmä: Ohjelman käyttäjä.

Kuvaus: Käyttäjä tekee testin. Kuvaruudulla näkyy kysymykset, joihin kaikkiin täytyy vastata. Kun vastaukset on syötetty, käyttäjä painaa Vastaa-painiketta.

Alkutila: Testiosio on avattu.

Skenaariot:

Syöte: Jokin vastauskenttä on tyhjä.
Tulos: Vastausta ei hyväksytä, näytetään virheilmoitus.

Syöte: Väärin muotoiltu vastaus.
Tulos: Vastausta ei hyväksytä, näytetään virheilmoitus.

Syöte: Kaikki vastaukset on muotoiltu oikein.
Tulos: Näytetään testin tulos.

Syöte: -
Tulos: Kysymyksessä näkyy vastauksen vaatima yksikkö.

Syöte: Testiosioon siirtyminen.
Tulos: Kysymykset ovat testiosion laskuja. Kysymysten lukumäärät ovat eri osa-alueilta asetusten mukaisesti.

Syöte: Jokin vastaus on väärin. Käyttäjä painaa "Vastaa-painiketta.
Tulos: Kuvaruudulla näkyy ilmoitus väärin menneistä vastauksista ja niiden oikeat vastaukset ja laskukaava.

Syöte: Testi on suoritettu hyväksytysti.
Tulos: Käyttäjällä on mahdollisuus tulostaa todistus.

6.3.5 Testiosion avaus

Sidosryhmä: Testin valvoja.

Kuvaus: Testin valvoja avaa testiosion testin suorittajan käyttöön.

Alkutila: Ohjelma on päävalikossa.

Skenaariot:

Syöte: Salasana on väärin.
Tulos: Ohjelma jää päävalikkoon, näytetään virheilmoitus.

Syöte: Salasana on oikein.
Tulos: Ohjelma siirtyy testiosioon.

6.3.6 Salasanan vaihtaminen

Sidosryhmä: Testin valvoja tai ohjelman ylläpitäjä.

Kuvaus: Ylläpitäjä vaihtaa salasanaa syöttämällä vanhan salasanan ja kaksi kertaa uuden salasanan.

Alkutila: Ylläpitäjä on kirjautunut ylläpito-osioon.

Skenaariot:

Syöte: Vanha salasana on väärin.
Tulos: Salasanaa ei vaihdeta.

Syöte: Uudessa salasanassa on kirjoitusvirhe.
Tulos: Salasanaa ei vaihdeta.

Syöte: Uusi salasana on tyhjä.
Tulos: Salasanaa ei vaihdeta.

Syöte: Salasanat ovat oikein.
Tulos: Uusi salasana hyväksytään.

Oikein muotoiltu salasana on 1-64 merkkiä pitkä.

6.3.7 Kysymyksen lisääminen

Sidosryhmä: Testin valvoja tai ohjelman ylläpitäjä.

Kuvaus: Ylläpitäjä lisää testi/harjoitustietokantaan kysymyksen.

Alkutila: Ylläpitäjä on kirjautunut ylläpito-osioon.

Skenaariot:

Syöte: Vastaus on väärin muotoiltu.
Tulos: Kysymystä ei lisätä, näytetään virheilmoitus.

Syöte: Kysymys
Tulos: Uusi kysymys tietokannassa.

Kysymykseltä vaaditut kentät:

Kenttä	Pituus
Laskutyyppi	vakio
Vaikeustaso	vakio
Lasku	1-1000
Vastaus	1-10
Yksikkö	0-20
Laskukaava	0-50
Vihje1	0-500
Vihje2	0-500

6.3.8 Kysymyksen muokkaaminen

Sidosryhmä: Testin valvoja tai ohjelman ylläpitäjä.

Kuvaus: Ylläpitäjä muokkaa olemassaolevaa kysymystä.

Alkutila: Ylläpitäjä on kirjautunut ylläpito-osioon.

Skenaariot:

Syöte: Vastaus on väärin muotoiltu.
Tulos: Kysymystä ei muokata, näytetään virheilmoitus.

6.3.9 Kysymyksen poistaminen

Sidosryhmä: Testin valvoja tai ohjelman ylläpitäjä.

Kuvaus: Ylläpitäjä valitsee poistettavat kysymykset ja painaa Poista-painiketta.

Alkutila: Ylläpitäjä on kirjautunut ylläpito-osioon.

Skenaariot:

Syöte: Valitut kysymykset.
Tulos: Kysymykset ovat poissa tietokannasta.

Testataan Poista-painike myös kun mitään ei ole valittu.

6.3.10 Läpipääsytodistuksen tulostaminen

Sidosryhmä: Testin tekijä tai valvoja.

Kuvaus: Ohjelma tulostaa läpipääsytodistuksen.

Alkutila: Testi on suoritettu onnistuneesti tai ylläpitäjä on kirjautunut ylläpito-osioon.

Skenaariot:

Syöte: Virheellinen nimi, sukunimi tai henkilötunnus.
Tulos: Tulostusta ei käynnistetä.

Syöte: Nimi, sukunimi, henkilötunnus oikein.
Tulos: Tulostetaan todistus.

Syöte: Käyttäjä painaa Tulosta-painiketta.
Tulos: Käyttäjälle näytetään tulostusdialogi jos tulostinta ei ole valittuna asetuksissa, muuten tulostetaan asetusten tulostimella.

Nimi ja henkilötunnus -kenttien pituudet ovat 1-100 merkkiä. Molemmat ovat pakollisia tulostusta varten.

6.3.11 Ylläpitotehtävät

Sidosryhmä: Testin valvoja tai ohjelman ylläpitäjä.

Kuvaus: Ylläpitäjä voi vaihtaa ohjelman parametreja: testin laskujen suhde, harjoittelujakson pituus, tietokantojen sijainnit. Tallennetaan Tallenna-painikkeesta.

Alkutila: Ylläpitäjä on kirjautunut ylläpito-osioon.

Skenaariot:

Syöte: Testin laskujen lukumäärien summa on 10.
Tulos: Muutokset tallennetaan.

Syöte: Testin laskujen lukumäärien summa ei ole 10.
Tulos: Muutoksia ei tallenneta, näytetään virheilmoitus.

Syöte: Harjoittelujakson pituus, positiivinen kokonaisluku.
Tulos: Muutokset tallennetaan.

Syöte: Testi/harjoittelutietokannan sijainti.
Tulos: Muutokset tallennetaan.

Syöte: -
Tulos: Laskut on tallennettu salattuna.

6.4 Vaatimusten validointi

Vaatimusten validoinnin tarkoitus on testata täyttääkö järjestelmä vaatimusdokumentin vaatimukset. Alla olevaan taulukkoon on koottu vaatimukset ja ne todentavat laajennetut käyttötapaukset. Vaatimusten numerot vastaavat vaatimusdokumentin numerointia ja käyttötapausten numerot vastaavat edellä listattuja laajennettuja käyttötapauksia.

6.4.1 Toiminnalliset vaatimukset

Vaatimus	Laajennettu käyttötapaus
1	6.3.1
2	6.3.2
3	6.3.2
4	6.3.2
5	6.3.2
6	6.3.2, 6.3.4
7	6.3.2
8	6.3.3
9	6.3.2
10	6.3.2
11	6.3.2
12	6.3.2
13	6.3.4
14	6.3.4
15	6.3.4
16	6.3.4, 6.3.10
17	6.3.6
18	6.3.7
19	6.3.11
20	6.3.8
21	6.3.9
22	6.3.10
23	6.3.11
24	6.3.11
25	6.3.10
26	Ei toteuteta.
27	6.3.11

6.4.2 Ei-toiminnalliset vaatimukset

Vaatimus	Validointi
1	Käyttötapausten virhetilanteiden testaus.
2	Käytettävyystesti.
3	Käytettävyystesti.
4	Käytettävyystesti.
5	Käytettävyystesti ja EUCT 6.3.5.

6.5 Hyväksymiskriteerit

Järjestelmätestaus on riittävä, kun käyttötapausten skenaariot on suoritettu onnistuneesti.

7 Muu testaus

Jos jää aikaa ja löytyy sopivia työkaluja, laajennettujen käyttötapausten testaus automatisoidaan skripteillä/makroilla. Lisäksi käyttöliittymän stabiilius voidaan testata jollain työkalulla, joka syöttää järjestelmälle suuren määrän hiiri/näppäin-tapahtumia. Kun logiikka- ja tietokantakomponentit on integroitu, yhdistelmää voisi testata ohjelmalla, joka kutsuu logiikan julkisia metodeja satunnaisesti. Tällä saattaisi löytää nopeasti tilanteita, joissa logiikka joutuu sisäisesti virheelliseen tilaan.

8 Testausaikataulu

Yksikkötestaus aloitetaan samaan aikaan toteutuksen kanssa ja jatketaan koko toteutuksen ajan. Integrointitestaus aloitetaan heti kun mahdollista ja saatetaan loppuun viikolla 14 (2.4-8.4). Järjestelmätestaus tehdään integrointitestauksen jälkeen viimeistään viikoilla 15 (9.4-15.4).