JouTo

Toimittajan työväline

Testaussuunnitelma

Versio 1.0

8.2.2001
Helsingin yliopisto
Tietojenkäsittelytieteen laitos
Ohjelmistotuotantoprojekti, kevät 2001, ryhmä 3

Projektiryhmä:	Petri Hellemaa (PH)
	Antti Koskinen (AK)
	Lari Laine (LL)
	Jarno Saarto (JS)
	Eija Teitto (ET)
	Harri Tuuloskoski (HT)
Ohjaaja:	Kim Viljanen
Vastuuhenkilö:	Raine Kauppinen
Asiakkaat:	Helena Ahonen-Myka
	Greger Lindén

Dokumentin versiohistoria

0.1	22.2.2001	Ensimmäinen versio
0.2	29.2.2001	Toinen versio, paljon tekstiä lisätty
0.3	04.2.2001	Lisätty tekstiä, tarkennettu kappaleita
0.4	12.3.2001	Katselmusta varten viimeistelty versio. Varsinaiset testitapaukset puuttuvat vielä
1.0	22.4.2001	Lopullinen versio

Tästä dokumentista vastaa Harri Tuuloskoski (HT).

Sisältö

1 Johdanto (HT)

Tässä testaussuunnitelmassa esitellään JouTo-ohjelmistotuotantoprojektin tuotteena syntyvälle työvälineelle suunniteltu tes

Tässä testaussunnitelmassa kuvataan JouTo-ohjelmistotuotantoprojektin tuotteena syntyvän ohjelmiston suunniteltu testaus ja testausvaiheen henkilöorganisointi. Lisäksi dokumentissa kuvataan testauksessa käytettävä ohjelmisto sekä kaikki testauksessa käytettävät testitapaukset, joiden perusteella ohjelmiston toimintaa ja oikeellisuutta testataan.
JouTo on lyhenne englannin kielen sanoista *"A Journalist's Tool"*.

Testauksen tarkoituksena on löytää ohjelmistosta mahdollisimman suuri osa virheistä ennen ohjelmiston julkaisua. Testaus on valitussa projektimallissa oma vaihe, joka suoritetaan varsinaisen toteutuksen jälkeen. Testaukseen osallistuu koko projektiryhmä.

Testauksen tuloksista kirjoitetaan testausdokumentti. Dokumentista nähdään ohjelmistolle suoritettu testaus sekä löydetyt virheet. Tämä dokumentti liitetään loppuraporttiin.

1.1 Testauksen kohde ja tavoitteet (ET)

Testauksen kohteena JouTo-ohjelmisto: asiakasohjelmisto, palvelinohjelmisto ja näiden välinen tietoliikenne.

Testauksella pyritään varmistumaan ohjelmistolle asetetuista laatuvaatimuksista. Ohjelmiston tulee täyttää asiakkaan kanssa määritellyt vaatimukset, jotka on kuvattu vaatimusmäärittelydokumentissa.
Tarkempi kuvaus testauksen kohteista ja tavoitteista löytyy kappaleesta 4

1.2 Määritelmät, termit ja lyhenteet

JouTo Tässä projektissa tuotettava ohjelmisto item[JTest] Parasoft-yhtiön kehittämä tuote Java-pohjainen testaustyökalu, jolla JouTo-ohjelmistoa testataan. Lisätietoja osoitteesta `http://www.parasoft.com`

1.3 Viitteet

[PRESS97] Software Engineering, 4. edition, Roger S. Pressman

1.4 Dokumentin rakenne

Luvussa 1 kerrotaan testausdokumentin tavoitteet ja yleiset asiat.

Luvussa 2 kerrotaan testausympäristöstä, testaukseen käytettävistä ohjelmista sekä luotavista testausluokista.

Luvussa 3 kerrotaan käytettävistä testimenetelmistä ja niiden käyttöalueesta sekä tarkoituksesta.

Luvussa 4 kerrotaan testauksen kohteista ja tavoitteista testauksen suhteen.

Luvussa 5 kuvataan testauksen organisointi ja tuloksista raportointi.

Luvussa 7 kuvataan testauksessa käytettävät testitapaukset.

Luvussa 8 tarkastellaan testausprosessia,testitapauksia, testaustuloksia sekä hyväksymisperusteita.

2 Testausympäristö

Testausympäristönä toimii Helsingin yliopiston tietojenkäsittelytieteen laitoksen Windows ja Linux-ympäristö.

2.1 Ohjelmisto (ET)

Testauksessa käytetään tietojenkäsittelylaitoksen laitteistoa. WWW-selaimen ikkunaan on ladattu Java Appletilla toteutettu käyttöliittymäsovellus. Varsinainen ohjelma on koodattu Javalla. Palvelimen ohjelmisto on toteutettu Servetillä. Palvelimella on Helsingin yliopiston tietojenkäsittelytieteen laitoksen Oracle-tietokanta.

2.2 Testimodulit

JouTo-ohjelmiston testauksessa luodaan testimoduleita niitä testattavia ominaisuuksia varten, joita ei JTest-ohjelmistolla (katso kappale 2.3) pystytä testaamaan. Testimodulien tarkoituksena on simuloida muuta ohjelmistoa, ja tutkia testattavan ohjelmiston osan toimivutta annetuilla syötteillä.

Tätä osaa ei voida vielä tarkentaa ennen varsinaisten testitapausten tiedostamista.

2.3 Apuvälineet

JouTo-ohjelmiston testauksessa käytetään valmista testausohjelmistoa testausta helpottamaan. Testiohjelmistona käytetään Parasoft Corp. tuottama JTest-ohjelmistoa. JTest-ohjelmistolla pystytään testaamaan automaattisesti seuraavat asiat

- Koodin ulkoasun oikeellisuus (White Box-testausta). Ohjelma tutkii ohjelmakoodin kirjoitusasun oikeellisuutta käytettyjen Java-standardien mukaan ja ilmoittaa virheistä. Tämä osuus on käyttäjän täysin muokattavissa, ja se muokataan käyttäjään JouTo-projektin Java Tyyliohjetta.
- Ohjelman sisäinen toiminta ja metodien looginen toiminta eri syötteillä. Ohjelma pystyy havaitsemaan mm. käyttämättömät muuttujat ja ohjelman osat. Ohjelma pystyy testaamaan mahdolliset käsittelemättömät keskeytykset ja ilmoittamaan ne aiheuttavat metodit. Tämän lisäksi ohjelma pystyy testaamaan metodien toiminnan eri parametreilla ja varoittamaan epäloogisista operaatioista ilmenneistä virheistä.
- Eri luokkien metodien ja funktioiden testaus (Black Box-testausta). Ohjelma pystyy automaattisesti tutkimaan ja testaamaan eri luokkien ja metodien tuloksia sekä toimintoja annetuilla syötteillä, ja etsimään mahdollisia virheitä toiminnoista.
- Luokkien välisten rajapintojen testaus. Ohjelma pystyy luomaan ilmentymiä luokista ja suorittamaan ohjelmaa simuloiden ajonaikaista suoritusta. Ohjelma pystyy testaamaan näin luotujen luokkien tarjoaman rajapinnan toimintaa eri syötteillä.

Ohjelma sisältää suuren joukon valmiita testitapauksia sekä testaussääntöjä. Valmiiden sääntöjen lisäksi joututaan ohjelmistoon syöttämään omia testitapauksia, jotta kaikki eri osa-alueet ja tilanteet tulevat testatuiksi.

3 Testauksen jako (HT)

Tässä kappaleessa kuvataan erilaiset testaustyylit joilla JouTo-ohjelmistoa testataan sekä kuvataan erilaiset testausmenetelmät, joita käytetään JouTo-ohjelmiston testauksessa.

3.1 Testausmenetelmät

JouTo-ohjelmiston testauksessa käytetään seuraavia testausmenetelmiä.

3.1.1 White-Box testaus

White-Box testauksessa pyritään löytämään ohjelmakoodista joitain puutteellisuuksia tai toimintalogiikan virheitä. Jokainen ohjelmoija tarkkailee aktiivisesti jo toteutusvaiheessa omaa ohjelmakoodiaan. Varsinaisessa White-Box - testausvaiheessa testataan tärkeimmät algoritmit sekä kriittisin osa-alue ohjelmakoodista.

White-box testauksessa pyritään saavuttamaan seuraavat asiat [PRESS97]

1. Modulin jokainen erillinen polku saavutaan ainakin kertaalleen ja myös jokaista ohjelman polkua käytetään jossain vaiheessa.
2. Jokaista modulin loogista operaattoria kokeillaan sekä tosi että epätosi - arvoilla.
3. Suoritetaan kaikki modulin silmukat sallituilla arvoilla.
4. Varmistetaan sisäisten tietoalkioiden- ja rakenteiden toimivuus ja oikeellisuus.

3.1.2 Black-Box testaus

Black-Box testaus keskittyy ohjelmiston toiminnallisiin vaatimuksiin. Black-Box testauksessa pyritään löytämään virheitä ohjelmiston rajapinnoista ja komponenttien välisestä toiminnasta. Se ei ota kantaa järjestelmän sisäiseen toimintaan.

Black-Box testauksessa pyritään löytämään seuraavanlaiset virheet [PRESS97]

1. Virheelliset tai puuttuvat metodit.
2. Virheet rajapinnoissa.
3. Virheet tietorakenteissa tai ulkoisen tietokannan käsittelyssä.
4. Virheet tai puutteet suorituskyvyssä.
5. Ohjelman alustus tai päätösvirheet.

3.1.3 Käytettävyystestaus

Käytettävyystestauksessa pyritään etsimään toimimattomia tai huonosti toimivia piirteitä ohjelmiston käyttöliittymästä tai muusta ohjelmiston käyttöön liittyvästä asiasta.
Käytettävyystestestaus on aistinvaraista testaamista, jossa testataan sekä käyttöliittymän toiminta ja käyttömukavuus että ohjelmiston yleinen käyttömukavuus loppukäyttäjän kannalta.

3.2 Modulitestaus

Modulitestauksella pyritään testaamaan erillisten modulien toimintaa ja näiden rajapintaa erilaisilla syötteillä.Tämä suoritetaan Black-Box -menetelmällä. Modulitestaukseen kuuluu myös osana modulin rakenteiden ja ohjelmakoodin logiikan testaaminen ohjelmakoodin tasolla. Tähän käytetään White-Box -menetelmää.

Black Box-testausta suoritetaan joko itse tehdyillä testausluokilla tai valmiilla testauksohjelmistoilla. Tulosteista ja virhetilainteista päätellään ohjelmiston toimivuus.
White Box-testaus suoritetaan ryhmän toimesta manuaalisesti tutkimalla valittu osa ohjelmakoodia ja kommentoimalla löytyneitä puutteita tai epäloogisia toimintoja.

Suurin osa modulitestauksesta tehtään JTest-ohjelmiston avulla (katso kappale 2.3). Ne osat joito JTest-ohjelmistolla ei pystytä testaamaan, testataan itsetehtyjen testausmoduleitten(Black Box-testaus) tai ryhmäläisten (White Box-testaus) toimesta.

3.3 Järjestelmätestaus

Järjestelmätestauksella pyritään testaamaan modulien välistä toimintaa ohjelmiston osana. Järjestelmätestauksessa annetaan järjestelmään (testattavista komponenteista koottuun osaan) syötteitä ja tutkitaan tulosteiden järkevyyttä sekä tiedon kulkua järjestelmässä. Samalla testataan ohjelmistoon asetettuja vaatimuksia (esimerkiksi nopeusvaatimuksia) järjestelmätasolla.

Osa järjestelmätestauksesta tehdään Jtest-ohjelmistolla (katso kappale 2.3). Luokkakokonaisuuksien rajapintojen ja sisäisten toimintojen testaus joudutaan kuitenkin tekemään käyttämällä itsetehtyjä testausmoduleita.

3.4 Käyttöliittymätestaus

Käyttöliittymätestauksessa pyritään toteamaan käyttöliittymän soveltuvuus asiakkaalle sekä todentamaan siihen kohdistetut vaatimukset ja toivomukset. Käyttöliittymätestauksessa pyritään löytämään liittymästä ja asiakkaan puolen toiminnoista sellaisia toimintoja jotka tekevät ohjelman käytön vaikeaksi tai hankalaksi. Käyttöliittymätestaus suoritetaan ensisijaisesti ryhmän sisällä. Ryhmän sisäisten testausten jälkeen pyritään syntynyt käyttöliittymä testaamaan myös asiakkailla jotta heiltä saadaan mielipiteet ja kommentit liittymästä.

4 Testauksen kohde ja tavoitteet (ET)

Testaussuunnitelmassa kuvataan kokonaisen, valmiin järjestelmän testaus. Moduulivastaavat huolehtivat moduulitestausten läpiviennistä ennen kuin moduleita yhdistellään joko toisiin moduleihin tai toiseen moduliin. Integrointitestaukset suoritetaan sekä koko ohjelmistolle että ohjelmiston eri kokonaisuuksille.

Tietoliikenteen mukaantulo tuo omat ominaisuudet ja toiminnot testaukseen. Testauksessa ei riitä, että testataan ainostaan uuden koodin osuus toimintojen oikeellisuudessa, vaan on myös testattava valmiiden komponenttien oikeellisuus JouTo-projektissa.

Testauksella pyritään varmistumaan ohjelmistolle asetetuista laatuvaatimuksista. Ohjelmiston tulee täyttää asiakkaan kanssa määritellyt vaatimukset, jotka on kuvattu vaatimusmäärittelydokumentissa. Ohjelman toimintojen tulee toimia dokumentissa kuvatulla tavalla. Lisäksi testisuunnitelmassa määritellään ohjelmiston toiminta virhetilanteissa.

Käyttöliittymän käytettävyystestaus suoritetaan ryhmän sisällä. Ennen testauksen hyväksymistä pyritään hankkimaan kommentteja myös asiakkailta.

Varsinaisen toiminnallisuuden lisäksi olevia ei-toiminnallisia ominaisuuksia ei testata. Ei-toiminnallisia ominaisuuksia ovat ainakin suorituskyky, siirrettävyys ja yhteensopivuus.

Testaussuunnitelman mukaisesti suoritetun testauksen ja mahdollisten virheiden korjauksen jälkeen tuote voidaan luovuttaa asiakkaalle.

5 Testauksen organisointi ja raportointi(HT)

Seuraavassa kappaleessa kerrotaan yleisesti testauksen organisoinnista ja raportoinnista.

5.1 Testauksen organisointi

Testausvaihe sijoittuu toteutusvaiheen jälkeen. Testaukseen osallistuu koko JouTo-projektin ryhmä, ja sitä johtaa vaiheen projektipäälikkö. Testausvaihe kestää kaksi viikkoa, ja siinä ajassa pyritään löytämään ja korjaamaan suurimmat virheet toiminnassa jä käyttöliittymässä.

5.2 Testauksen henkilöjärjestelyt

Testausvaiheesta vastaa vaiheen projektipäällikkö. Testaus suoritetaan suunniteltujen testitapausten perusteella. Testit suorittaa joko modulin tai ohjelmaosuuden luoja, tai projektipäälikön osoittama henkilö.

Black Box-testauksessa määrätty testaaja testaa ohjelman osaa, ja kirjoittaa tulokset testaussuunnitelmaan. Mikäli toiminta on virheellinen tai puutteellinen, korjaukset ohjelmaan tekee ohjelmakoodin kirjoittaja tai projektipäälikön osoittama henkilö.

White Box-testauksessa määrätään osa ryhmän jäsenistä tutkimaan määrätty osa ohjelmakoodista tarkoituksenö löytää jokin virhe ohjelman toimintalogiikassa tai varsinaisessa toiminnassa. Osatulokset kootaan yhteen testauskatselmuksessa, jonka perusteella osuuken ohjelmoinut henkilö korjaa lähdekoodinsa.

Testauksessa testivalineenä käytetään joko JTest-ohjelmistoa (katso kappale 2.3) tai JouTo-projektissa ohjelmoituja testimoduleita.

5.3 Testauksen kulku

Etukäteen testaussuunnitelmassa määritelty testitapaus suoritetaan ja sen tulos kirjataan testausdokumenttiin. Mikäli testauksen tulos ei vastaa toivottua tilannetta, eli ohjelmiston sen osan toiminnassa on virhe, kyseisen osan ohjelmoinut henkilö korjaa virheen annetun testitapauksen ja tuloksen perusteella.

Ohjelmiston testattavat osat jaetaan vaiheen projektipäälikön toimesta.
Testauksen tulokset jaetaan kolmeen (3) eri vaarallisuusluokkaan

1	Fataali	Bugi joka johtaa ohjelman kaatumiseen tai ohjelman täydelliseen toimimattomuuteen
2	Vakava	Bugi joka estää ohjelman normaalin toiminnan, tai toimii aivan eri tavalla kun on tarkoitettu
3	Pieni	Bugi jonka takia ohjelman toiminnassa on pieniä puuteita, muttei ohjelman barsinainen toiminta esty

Testauksessa pystytään etenemään kyseisen modulin, luokan tai luokkaryhmän osalta vasta kun tason 1 ja 2 virheet on korjattu. Tason 3 virheet korjataan projektipäälikön ja ryhmän harkinnan mukaan ajan puitteissa.

Testauksen järjestys on seuraava

1. Luokkien testaus Black Box menetelmällä
2. Määrättyjen luokkien testaus White Box-menetelmässl

3. Luokkaryhmien integrointitestaus (osa Järjestelmätestausta)

4. Luokkaryhmien välisen toiminnan testaus (varsinainen järjestelmätestaus)

5. Koko ohjelman toiminnan testaaminen kokonaisuutena

6. Käyttöliittymätestaus (vasta kun edelliset kohdat on testattu)

5.4 Ominaisuudet joita ei testata

Kaikki ominaisuuksia ei voida testata. Testataan ohjelmiston jä käyttäjän kannalta tärkeimmät. Nämä on jaettu kahteen kategoriaan: toiminnallisiis ja ei-toiminnallisiin ominaisuuksiin.

5.4.1 Toiminnalliset ominaisuudet

Seuraavia toiminnallisia ominaisuuksia ei testata:

- Monen asiakasohjelmiston rinnakkainen samanaikainen toiminta
- Tietoliikenneverkosta johtuvat tiedonsiirron virhetilanteet
- Käyttöliittymän sopivuutta loppukäyttäjälle
- Epärealistiset, ohjelmistosta riippumattomat virhetilanteet asiakas- tai palvelinohjelmistossa tai näiden välisessä tietoliikenteessä.

5.4.2 Ei-toiminnalliset ominaisuudet

Seuraavia ei-toiminnallisia ominaisuuksia ei testata:

- Suorituskyky
- Siirrettävyys
- Yhteensopivuus

6 Testauksen tehtäväjärjestys ja testausmenettely (HT,ET)

Tämä kappale kertoo testauksen tehtäväjärjestyksen, käytettävät menetelmät kullakin osalla sekä testausjärjestyksen.

6.1 Testattavat osat

Tässä kappaleessa kuvataan eri osien testausmenetelmät.

Modulitestaus

Modulitestaukseen osallistuu jokainen luokka.
White Box-testaukseen osallistuvat luokat

- UIManager
- Treemanager

Black Box-testaukseen osallistuvat kaikki luokat

Järjestelmätestaus

Järjestelmätestaus suoritetaan eri luokkaryhmille. Siihen osallistuvat seuraavat seuraavat luokkaryhmät:

Käyttöliittymäluokat

- UIManager
- ClipboardManager
- valmiista komponenteista luodut ilmentymät
- muut liittyvät luokat

XML-puun rakennus- ja muokkausluokat

- Treemanager
- Node
- SAXParser
- DTDParser
- muut liittyvät luokat

Tietoliikenne ja tiedon tallennus-luokat

- Comminications
- Servlet
- DB
- muut liittyvät luokat

Käyttöliittymätestaus

Käyttöliittymätestaukseen osallistuvat asiakasohjelmiston käyttöliittymäkomponentit. Käyttöliittymätestaus totetutetaan suorittamalla annettuja tehtäviä ja tutkimalla kuinka käyttöliittymä toimii kyseisten tehtävien suorittamisessa.

Muut

Tähän tulee lista muista testattavista osista.

Tällä hetkellä muita testattavia osia ei ohjelmistossa ole.

6.2 Tehtäväjärjestys

Suoritettavan ohjelmiston testauksen eri osien tärkeysjärjestys:

1. Ohjelmoijan henkilökohtainen laadunvalvonta ja esitestaus.
2. Eri modulien Black Box-testaus.
3. Järjestelmätestaus.
4. Käyttöliittymätestaus
5. Eri modulien White Box-testaus.
6. Muu mahdollinen testaus.

Testauksessa pyritään testaamaan ohjelma kokonaisuudessaan. Yllä olevaa listaa käytetään priorisoimaan testausta ajanpuutteen tms. tapauksessa.

7 Toimintojen testitapaukset

Tänne tulee kuvaus testitapauksista. Varsinaiset testitapaukset laitetaan liitteinä dokumenttiin.

Testitapaukset jakautuvat neljään (4) ryhmään:

1. Eri luokille tehtävät Black Box-testaukset (Liite

8 Testauksen hyväksyntä

Testaus katsotaan hyväksytyksi, kun seuraavat vaiheet on suoritettu:

- White Box-testaus on suoritettu määrättyjen luokkien osalta ja löydetyt virheet on korjattu.
- Black Box-testaus on suoritettu jokaisen luokan jokaiseen ulkoisen rajapinnan metodiin ja löydetyt virheet on korjattu.
- Järjestelmätestaus on suoritettu määrättyihin luokkakokonaisuuksiin ja kaikki fataalit virheet on korjattu.
- Käyttöliittymätesti on suoritettu kertaalleen ryhmän sisällä.

Ohjelman on toimittaa normaalissa käytössä ilman näkyviä virheitä ja kaikkien toteutettujen ominaisuuksien on toimittava määritellyllä tavalla.

Ohjelman testaus voidaan katsoa hyväksytyksi myös vaikka ym. ominaisuuden eivät täyttyisi mikäli JouTo-projektin ryhmä tai sen vastuuhenkilön päättävät asiasta.

A Liite: Luokkien Black Box - testaukset

B Liite: White Box - testaukset

C Liite: Integrointitestauksen testitapaukset

D Liite: Käyttöliittymätestauksen testitapaukset