Helsingin yliopisto

Tietojenkäsittelytieteen laitos

Ohjelmistotuotantoprojekti XPerf

Testaussuunnitelma

Tommi Koivula
Antti Levomäki
Juha Mondolin
Timo Suomela

Versio 1.0
28. maaliskuuta 2003

Versiohistoria

Versio	Päivämäärä	Kommentti / muutokset	Tekijä
0.1	17.3.2003	Runko	Juha Mondolin
0.2	19.3.2003	Pääkohdat, osa testitapauksista	Juha Mondolin
0.3	20.3.2003	Testitapauksia lisätty	Tommi Koivula Juha Mondolin
0.4	23.3.2003	Testitapauksia lisätty	Juha Mondolin
0.5	23.3.2003	Testitapauksia lisätty	Timo Suomela Annti Levomäki Tommi Koivula Juha Mondolin
1.0	28.3.2003	Viimeistelty valmis versio.	Tommi Koivula

Sisällys

1. Johdanto

Tämä dokumentti käsittelee XPerf-ohjelmistotuotantoprojektin testauksessa käytettäviä periaatteita sekä selvittää ne pelisäännöt, jotka koskevat valmiin ohjelmakoodin hyväksyntää. Testaussuunnitelma on jaettu viiteen eri osaan:

- "Testauksen kohde ja tavoitteet" kertoo mitä testataan ja mitä testauksella saavutetaan.

- "Testausympäristö" selvittää testauksessa käytettävät laitteistot ja ohjelmistot.

- "Testauksen organisointi ja raportointi" kertoo miten, mitä ja milloin testataan ja kuinka testeistä raportoidaan.

- "Testien hyväksymiskriteerit" kertoo miten testattu osa hyväksytään.

- "Testitapaukset" kuvaa ohjelman eri osiin liittyvät testit.

Varsinainen testaus on jaettu kolmeen erilliseen vaiheeseen siten, että *moduulitestaus* suoritetaan yksittäisten ohjelman osien ohjelmoinnin yhteydessä, tarkoituksena on jo tässä vaiheessa varmistaa mahdollisimman virheetön koodi. *Integraatiotestaus* suoritetaan moduuleita yhdistettäessä, sen tarkoituksena on varmistaa moduulien oikea yhteistoiminta. *Validointitestaus* on valmiin järjestelmän testausta vaatimusmäärittelyä[3] vastaan, sillä varmistetaan että ohjelmisto täyttää asiakkaan sille asettamat vaatimukset.

2. Testauksen kohde ja tavoitteet

Testauksen kohteena on MAISA-ohjelmistoon[1] tehty XMI-dokumenttien lukujärjestelmä. Testauksen tavoitteena on varmistua siitä että tuotettu ohjelmisto toimii oikein.

3. Testausympäristö ja testausmateriaali

Testausympäristönä on moduulitestuksen osalta moduulin ohjelmoijan oma laiteympäristö sekä Java 1.4.1 ja JUnit testausympäristö[2]. Integraatiotestaus suoritetaan TKTL:n laitoksen laiteympäristössä.

Testimateriaalit (testiluokat, JUnit-testit ja käytetyt testitiedostot) on *test*-pakkauksessa. Testauksen päätyttyä ne pakataan *test.zip* pakettiin, johon lisätään myös testaussuunnitelma ja testausdokumentti.

4. Testauksen organisointi ja raportointi

Testaus on organisoitu kolmeen eri vaiheeseen, jotka on tarkemmin esitelty alla.

4.1. Moduulitestaus

Moduulitestausta suorittaa kyseisen moduulin ohjelmoija. Moduulin sisäisen luokan ohjelmoija luo itse testiaineistonsa ja testaa luokan itse. Testausperiaatteena on siis white-box -testaus. Jokaisesta moduulista tehdään JUnit test-case.

Jokaista moduulia testataan ainakin seuraavasti:

- Moduulin logiikkaa testataan haarakattavasti käymällä läpi kaikki moduulin läpi vievät erilaiset polut. Testiajurit luodaan tarpeen mukaan.

- Paikallisten tietorakenteiden toimivuus testataan.

- Tarkistetaan ehtolauseet ja silmukat sekä näiden reunaehdot.

4.2. Integraatiotestaus

Integraatiotestaus alkaa kun moduulit liitetään MAISA-järjestelmään. Testauksen tarkoituksena on varmistaa moduulien oikea toiminta keskenään sekä MAISA-järjestelmän kanssa. Käytännössä integraatiotestausta suoritetaan toteutusvaiheen lopussa. Integraatiotestauksessa painotetaan XMI-lukijan testausta.

Tehdyistä testeistä raportoidaan käyttämällä testitapahtumalomaketta[Liite A].

4.3. Validointitestaus

Validointitestausta suoritetaan integraatiotestauksen jälkeen, tarkoituksena on varmistaa että toteutettu ohjelmisto täyttää asiakkaan vaatimukset. Käytännössä validointitestaus suoritetaan black-box testien avulla käyttämällä MAISA-järjestelmää. Testeissä pyritään varmistumaan MAISA-järjestelmän oikeasta toiminnasta toteutetun lukijamoduulin kanssa. Lisäksi kiinnitetään erityistä huomiota käytettävyyteen sekä ohjelmiston suorituskykyyn tehtyjen muutoksien jälkeen.

Testiaineistona käytetään itse tehtyjä ArgoUML-työkalulla tehtyjä kaavioita ja asiakkaan toimittamia Rational Rose –työkalulla tehtyjä kaavioita. Lisäksi käytetään valmiita Prolog-testiaineistoja, joilla Perf+ -projektissa MAISA-järjestelmää on aiemmin testattu.

5. Testien hyväksymiskriteerit

Testattujen toimintojen hyväksymismenettelystä: toiminto hyväksytään, kun annetut testitapaukset on suoritettu virheittä tai mikäli virheitä on ilmennyt, ne on korjattu ja testaus on suoritettu virheittä tämän jälkeen.

Mikäli testeissä löytyy MAISA-järjestelmästä virheitä jotka eivät liity tuotettavaan ohjelmaan nämä virheet raportoidaan eteenpäin MAISA-järjestelmän asiantuntija Juha Gustafssonille.

6. Testitapaukset

Tässä luvussa on lueteltu ryhmiteltynä tehtävät testitapaukset tunnuksineen. Testauksen kulusta raportoivaan testausdokumenttiin merkitään aina tehdyistä testeistä vastaavien testitapauksien tunnukset.

6.1. Lukijan valintajärjestelmä

valinta-1
Tarkoitus: Lukijamoduulin valintajärjestelmän testaaminen
Kuvaus: Annetaan MAISA_systemille jokin sellainen tiedosto luettavaksi joka saa kaikkien lukijoiden canRead()-metodin palauttamaan FALSEn.
Olosuhteet:
Odotettu tulos:

valinta-2
Tarkoitus: Lukijamoduulin valintajärjestelmän testaaminen
Kuvaus: Annetaan MAISA_systemille jokin sellainen tiedosto luettavaksi joka saa jonkin lukijan canRead()-metodin palauttamaan TRUEn.
Olosuhteet:
Odotettu tulos:

valinta-3
Tarkoitus: Lukijamoduulin luonnin testaaminen
Kuvaus: Lisätään lukijamoduulien konfiguraatiotiedostoon moduleita jotka: 1. Eivät toteuta DocumentReader-rajapintaa 2. Eivät ole olemassa 3. Löytyvät ja toteuttavat DocumentReader-rajapinnan
Olosuhteet:
Odotettu tulos: Kohdat 1-2 aiheuttavat keskeytyksen, kohta 3 ei aiheuta keskeytystä.

6.2. PrologDocumentReader

prolog-1
Tarkoitus: Prolog-lukijamoduulin toiminnan virheiden löytäminen
Kuvaus: Tehdään ajuri, jolla voidaan lukea sekä Prolog-lukijaa että XMI-lukijaa käyttäen. MAISA-järjestelmän tietorakenteisiin lisätään metodit tietojen tulostamista varten. Jo olemassaolevasta Prolog-testiaineistosta (osasta) tehdään vastaavat XMI-dokumentit ja molemmat ajetaan ajurilla, joka vertailee syntyneiden tietorakenteiden eroavaisuuksia.
Olosuhteet: Ajuriluokkana toimii *Ti_prolog-1* –luokka.
Odotettu tulos: Merkittäviä eroavaisuuksia XMI:n ja Prologin välillä ei löydy.

6.3. XMI-Parser

T_parser-1
Tarkoitus: XMI-dokumenttien virheiden havaitseminen
Kuvaus: Luodaan XMI-jäsennin, joka yrittää lukea dokumentista 1. yhden elementin, jota dokumentissa ei ole yhtään. 2. monta samanlaista elementtiä, joita dokumentissä ei ole yhtään. 3. täsmälleen yhden elementin, jota dokumentissa on monta. 4. täsmälleen yhden elementin, jota on dokumentissa vain yksi. 5. monta samanlaista elementtiä, joita on dokumentissa useita. 6. monta samanlaista elementtiä, joita on dokumentissa vain yksi.
Olosuhteet: Junit-testausympäristö. Käytettävissä yo. mukainen XMI-dokumentti.
Odotettu tulos: Kohdat 1-3 aiheuttavat *XMIParseException*-poikkeuksen. Kohdat 4-6 eivät aiheuta poikkeusta.

T_parser-2
Tarkoitus: *NodeActionListener*-luokan tapahtumien oikea tapahtuminen.
Kuvaus: Luodaan XMI-jäsennin, joka lukee dokumentista juuri-elementin (*ModelElement*) ja sen lapsielementit. Lapsielementeillä on jokaisella eri prioriteetit. Testataan, että elementit luetaan oikeassa järjestyksessä.
Olosuhteet: JUnit-testausympäristö. Käytettävissä yo. mukainen XMI-dokumentti.
Odotettu tulos: Tiedot luetaan oikeassa järjestyksessä.

T_parser-3
Tarkoitus: Elementtien poissuodatuksen toimivuus**.**
Kuvaus: Jäsennetään XMI-dokumentti, jossa on poissuodatettavia elementtejä 1. niin, että ne sisältävät toisia poissuodatettavia elementtejä. 2. niin, että ne sisältävät elementtejä, joita ei suodateta. 3. jotka ovat tyhjiä. 4. jotka sisältyvät johon ei suodatettavaan elementtiin. Jäsennetään yo. vastaava dokumentti, jossa ei ole poissuodatettavia elementtejä. Tuloksena verrataan molempien jäsennyksien tulostamia tietoja.
Olosuhteet: JUnit-testausympäristö. Käytettävissä yo. mukaiset XMI-dokumentit.
Odotettu tulos: Vertailussa ei löydy eroavaisuuksia.

6.4. XMIDocumentReader

T_xmireader-1
Tarkoitus: XMI-lukijamoduulin virheenkäsittelyn testaaminen
Kuvaus: Annetaan lukijalle tiedosto josta on annettu DTD ja joka 1. on versioltaan XMI 1.0, mutta jonka sisältö ei ole annetun kieliopin mukainen. 2. ei ole XML-kielen kannalta hyvinmuodostettu. 3. Dokumentti ei sisällä kaikkia pakollisia elementtejä.
Olosuhteet:
Odotettu tulos: Kaikki kohdat aiheuttavat *MaisaParseException*-poikkeuksen.

T_xmireader-2
Tarkoitus: Kaaviota vastaavan lukijavalinnan virheidenkäsittely.
Kuvaus: Lukijalle annetaan syötteenä dokumentti, joka 1. sisältää kaikkia kaaviotyyppejä. 2. sisältää yhden tai useamman kaaviotyypin. 3. sisältää sekä ei-tuettuja että tuettuja kaaviotyyppejä 4. sisältää vain ei-tuettuja kaaviotyyppejä. 5. ei sisällä yhtään kaaviotyyppiä.
Olosuhteet: JUnit testaus.
Odotettu tulos: Kohdat 1-3 eivät aiheita poikkeusta ja kohdat 4-5 aiheuttavat *MaisaParseException*-poikkeuksen.

T_xmireader-3
Tarkoitus: Ulkopuolisten virheiden käsittely ja huomaaminen.
Kuvaus: 1. Lukijalle annetaan syötteenä tekstivirta, joka aiheuttaa *IOException*-poikkeuksen. 2. Jäsennintä ei löydy.
Olosuhteet: Junit testaus.
Odotettu tulos: Kaikissa tapauksissa jäsentäminen keskeytyy ja heitetään poikkeus.

Sama kuin testi **T_prolog-1** (Kts. luku 6.2).

6.5. Kaaviot

kaavioluku-1
Tarkoitus: Luokkakaavion sisältävän PROLOG-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen PROLOG-tiedosto joka sisältää luokkakaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät PROLOG-tiedoston sisällön kanssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut PrologDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-2
Tarkoitus: Luokkakaavion sisältävän XMI-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen XMI-tiedosto joka sisältää luokkakaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät XMI-tiedoston sisällön kanssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut XMIDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-3
Tarkoitus: Tilakaavion sisältävän PROLOG-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen PROLOG-tiedosto joka sisältää tilakaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät PROLOG-tiedoston sisällön kanssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut PrologDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-4
Tarkoitus: Tilakaavion sisältävän XMI-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen XMI-tiedosto joka sisältää tilakaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät XMI-tiedoston sisällön kannssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut XMIDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-5
Tarkoitus: Yhteistyökaavion sisältävän PROLOG-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen PROLOG-tiedosto joka sisältää yhteistyökaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät PROLOG-tiedoston sisällön kannssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut PrologDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: : MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-6
Tarkoitus: Yhteistyökaavion sisältävän XMI-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen XMI-tiedosto joka sisältää yhteistyökaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät XMI-tiedoston sisällön kannssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut XMIDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-7
Tarkoitus: Sekvenssikaavion sisältävän PROLOG-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen PROLOG-tiedosto joka sisältää sekvenssikaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät PROLOG-tiedoston sisällön kannssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut PrologDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-8
Tarkoitus: Sekvenssikaavion sisältävän XMI-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen XMI-tiedosto joka sisältää sekvenssikaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät XMI-tiedoston sisällön kannssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut XMIDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-9
Tarkoitus: Aktiviteettikaavion sisältävän PROLOG-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen PROLOG-tiedosto joka sisältää aktiviteettikaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät PROLOG-tiedoston sisällön kanssa.
Olosuhteet: : CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut PrologDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

kaavioluku-1
Tarkoitus: Aktiviteettikaavion sisältävän XMI-tiedoston lukemisen testaus.
Kuvaus: Avataan jokin sellainen XMI-tiedosto joka sisältää aktiviteettikaavion. Todetaan että lukemisen jälkeen MAISA-järjestelmän tietorakenteet täsmäävät XMI-tiedoston sisällön kannssa.
Olosuhteet: CDiagram luokkaan lisätään metodi printDiagram() joka tulostaa kyseisen kaavion kaikki elementit. DocumentReaderFactory on konfiguroitu DocumentReader-toteutuksella joka delegoi kaikki metodikutsut XMIDocumentReader instanssille ja read()-kutsun jälkeen tulostaa luetun kaavion kutsumalla sen printDiagram() metodia.
Odotettu tulos: MAISA-järjestelmän tietorakenteiden tietosisältö täsmää luetun tiedoston sisällön kanssa.

6.6. Indikaattori

ind-1
Tarkoitus: Indikaattorin testaus.
Kuvaus: Avataan jokin sellainen tiedosto jonka jäsentämiseen mennee alle kolme sekuntia.
Olosuhteet: DocumentReaderFactory on konfiguroitu yhdellä DocumentReader-toteutuksella jolla on seuraavanlaiset ominaisuudet: - supportsCancel() palauttaa aina FALSE. - canRead() palautta aina TRUE. - read(BufferedReader, PrintWriter, CProject) metodi odottaa kahden sekunnin verran ennenkuin palaa.
Odotettu tulos: Indikaattori ei näy käyttäjälle missään vaihessa.

ind-2
Tarkoitus: Indikaattorin testaus.
Kuvaus: Avataan jokin sellainen tiedosto jonka jäsentämiseen mennee yli kolme sekuntia.
Olosuhteet: DocumentReaderFactory on konfiguroitu yhdellä DocumentReader-toteutuksella jolla on seuraavanlaiset ominaisuudet: - supportsCancel() palauttaa aina FALSE. - canRead() palautta aina TRUE. - read(BufferedReader, PrintWriter, CProject) metodi odottaa kymmenen sekunnin verran ennenkuin palaa.
Odotettu tulos: : Indikaattori näkyy käyttäjälle kun noin kolme sekunttia on kulunut tiedoston valinnasta. Indikaattori näkyy noin seitsemän sekunttia, jonka aikana MAISA-järjestelmän pääikkuna on passivoituna, ennenkuin se sulkeutuu. Indikaattorissa ei ole 'Cancel' (Peruuta) näppäintä.

ind-3
Tarkoitus: Indikaattorin testaus.
Kuvaus: Avataan jokin sellainen tiedosto jonka jäsentämiseen mennee yli kolme sekuntia.
Olosuhteet Avataan jokin sellainen tiedosto jonka jäsentämiseen mennee yli kolme sekunttia. Jäsentäminen keskeytetään painamalla indikaattorin ' Cancel' (Peruuta) näppäintä. Olosuhteet: DocumentReaderFactory on konfiguroitu yhdellä DocumentReader-toteutuksella jolla on seuraavanlaiset ominaisuudet: - supportsCancel() palauttaa aina TRUE. - canRead() palautta aina TRUE. - read(BufferedReader, PrintWriter, CProject) metodi odottaa kymmenen sekunnin verran ennenkuin palaa.
Odotettu tulos: Indikaattori näkyy käyttäjälle kun noin kolme sekunttia on kulunut tiedoston valinnasta. MAISA-järjestelmän pääikkuna on passivoituna. Kun painetaan indikaattorin ' Cancel' (Peruuta) nappia niin indikaattori sulkeutuu ja MAISA-järjestelmän pääikkunaa ei ole enään passivoituna..

Lähteet

[1] MAISA, Metrics for Analysis and Improvement of Software Architectures http://www.cs.helsinki.fi/group/maisa/

[2] JUnit, Copyright © 1997-2002 JUnit.org, [http://www.junit.org/]

[3] Xperf –ohjelmistotuotantoprojektin määrittelydokumentti, [http://www.cs.helsinki.fi/group/maisa/xperf/doc/Maarittelydokumentti_1.1.6.PDF]

Liite A – Testaustapahtumalomake

Testin tunnus :
Päiväys :
Testaaja :
Oliko testitulos odotetun kaltainen (K/E) :
Saatu testitulos : (saatu tuloste, tai vastaava)
Testiolosuhteet : (testiin vaikuttaneita seikkoja, jotka eivät selviä kuvauksesta)
Käytetty testimateriaali :
Jatkotoimenpiteet : (jos sellaisia tarvitaan)