Testausraportti

Ohjelmistotuotantoprojekti Nero

Helsinki 16.12.2004

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (6 ov)

Projektiryhmä

Johannes Kuusela
Jyrki Muukkonen
Teemu Sjöblom
Ville Sundberg
Osma Suominen
Timi Tuohenmaa

Asiakas

Reijo Siven
Juhani Haavisto

Johtoryhmä

Juha Taina
Mikko Olin

Kotisivu

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

Versiohistoria

Versio	Päiväys	Tehdyt muutokset	Kirjoittaja
1.0	15.12.2004	Ensimmäinen ja viimeinen versio	Jyrki Muukkonen

Sisältö

1 Johdanto

Jo etukäteen oli tiedossa, että ohjelmistotuotantoprojektin aikataulu on tiukka. Toteutusvaiheessa ajanpuutetta korvattiin karsimalla testien kehittämisestä. Niinpä yksikkötestit toteutettiinkin pääosin jälkikäteen, eikä ennen varsinaista koodaamista kuten testaussuunnitelmassa [Ner04c] todettiin.

DbUnit[1] testeistä luovuttiin, lähinnä tietokantaan liittyvien rajoitusten ja kattavan ja oikeellisen testimateriaalin läpikäynnin työläyden takia. Niinpä tietokantaan liittyvien asioiden testaaminen jäi lähinnä kehityksen yhteydessä tehtävän kokeilun ja varsinaisen ohjelman käytön varaan.

Yksikkötestaus toteutettiin siinä määrin kuin DbUnit-testien puute sen mahdollisti. Integraatiotestaus siirtyi osaksi yksikkötestausta, koska se keskittyi vain yhteen Java-luokkaan. Myöskään rasitustestausta ei voitu suorittaa suunnitellulla tavalla DbUnitin puutteen takia. Tämä kuitenkin katsotaan korvatuksi sillä, että käytössä oli koko ajan oikea tietokanta, joka sisälsi mm. yli tuhat henkilöä sopimustietoineen. Järjestelmätestaus suoritettiin testaussuunnitelmassa kuvatulla tavalla. Hyväksymistestaus katsottiin kuitatuksi demotilaisuudella, sekä asentamalla ohjelmisto asiakkaiden koneille. Samalla annettiin myös pienehkö käyttökoulutus.

Yhteensä testaukseen liittyviksi merkattiin noin kuusikymmentä työtuntia. Projektisuunnitelmassa testaukseen oli varattu 220 tuntia. Toisaalta suunnitteluvaiheen ja toteutusvaiheen alun testaukseen liittyviä tunteja ei ole laskettu kyseiseen tuntimäärään mukaan. Varsinainen testaukseen liittyvä tuntimäärä on arviolta noin sata tuntia.

2 Yksikkötestaus

Suoritettu yksikkötestaus ei ollut niin kattavaa kuin oli alunperin tarkoitus. Tämä johtui siitä, ettei DbUnit-testausympäristöä saatu pystytettyä tarpeeksi helposti. DbUnit vaihtaa tietokannan tilaa oman suorituksensa ajaksi, joka olisi vaatinut omaa tietokantaa testejä varten. Lisäksi olisi pitäny luoda suuri määrä oikeellista testimateriaalia joka olisi pysynyt konsistentissa tilassa läpi projektin. Myöskin taistelu Oraclen kanssa oli aikaavievää, ja haaveet DbUnit-testeistä päätettiin haudata.

Yksikkötestejä ei siis suoritettu *NeroDatabase*-tietokantaluokalle eikä sitä kutsuville muiden luokkien metodeille. Lisäksi yksikkötestejä ei toteutettu *fi.helsinki.cs.nero.ui*-käyttöliittymäpakkauksen luokille. Näitä testattiin kuitenkin järjestelmätestauksessa.

Yksikkötestauksessa löydettyjä bugeja ei ole erikseen listattu, koska ne olivat yleensä pieniä kokonaisuuksia jotka korjattiin välittömästi virheen löydyttyä.

Yksikkötestauksen yhteydessä käytettiin lausekattavuuden mittaamiseen EMMA-työkalua[2]. EMMA raportoi luokka-, metodi-, lause- ja rivikattavuudet sekä oli suurena apuna testejä kirjoitettaessa. Yksikkötestattujen luokkien mitatut kattavuudet löytyvät taulukosta 1.

[1] `http://dbunit.sourceforge.net/`
[2] `http://emma.sourceforge.net/`

Luokka	**Metodikattavuus**		**Lausekattavuus**		**Rivikattavuus**	
Contract	10/10	100%	101/101	100%	23/23	100%
Person	8/9	89%	*80/293*	*27%*	*25/65*	*38%*
PhoneNumber	7/7	100%	80/80	100%	22/22	100%
Post	8/11	73%	*76/188*	*40%*	*23/52*	*44%*
Project	7/7	100%	61/61	100%	17/17	100%
Reservation	11/11	100%	118/118	100%	33/33	100%
Room	12/12	100%	118/143	83%	*39.8/50*	*80%*
TimeSlice	17/17	100%	193/193	100%	39/39	100%
Session	*28/50*	*56%*	*356/874*	*41%*	*106/220*	*48%*
NeroObserverManager	3/3	100%	68/68	100%	17/17	100%
Yhteensä	111/137	81%	*1251/2119*	*59%*	*344.8/538*	*64%*

Taulukko 1: Yksikkötestattujen luokkien testiajojen lausekattavuudet

2.1 TestContract

Yksikkötestit *fi.helsinki.cs.nero.data.Contract*-luokalle. Testeillä pystyttiin kattamaan kaikki Contract-luokan metodit, rivit sekä lauseet.

- **public void testContract()**
  Testit *Contract*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*
- **public void testGetContractID()**
  Testit *getContractID()*-metodille. *Testit läpäisty.*
- **public void testGetTimeSlice()**
  Testit *getTimeSlice()*-metodille. *Testit läpäisty.*
- **public void testWorkingPercentage()**
  Testit *workingPercentage()*-metodille. *Testit läpäisty.*
- **public void testGetTitle()**
  Testit *getTitle()*-metodille. *Testit läpäisty.*
- **public void testCompareTo()**
  Testit *compareTo(Object o)*-metodille. *Testit läpäisty.*
- **public void testCompareStartDates()**
  Testit *compare(Contract other)*-metodille. *Testit läpäisty.*
- **public void testGetPerson()**
  Testit *getPerson()*-metodille. *Testit läpäisty.*
- **public void testGetProject()**
  Testit *getProject()*-metodille. *Testit läpäisty.*

- **public void testToString()**
  Testit *getToString()*-metodille. *Testit läpäisty.*

2.2 TestPerson

Yksikkötestit *fi.helsinki.cs.nero.data.Person*-luokalle. Testeillä pystyttiin peittämään kaikki metodit lukuunottamatta tietokantaa vaativia metodeita, kuten *getStatus()* ja *contractBetweenDates()*. Koska kyseiset metodi on myös selvästi *Person*-luokan suurimmat metodit, jäi lausekattavuus erittäin alhaiseksi (27%).

- **public void testPerson()**
  Testit *Person*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*
- **public void testGetPersonID()**
  Testit *getPersonID()*-metodille. *Testit läpäisty.*
- **public void testGetName()**
  Testit *getName()*-metodille. *Testit läpäisty.*
- **public void testGetContracts()**
  Testit *getContracts()*-metodille. Testi ei tarpeeksi kattava, metodi käyttää tietokantaa. *Testit läpäisty.*
- **public void testGetReservations()**
  Testit *getReservations()*-metodille. Testi ei tarpeeksi kattava, metodi käyttää tietokantaa. *Testit läpäisty.*
- **public void testToString()**
  Testit *toString()*-metodille. *Testit läpäisty.*
- **public void testCompareTo()**
  Testit *compareTo()*-metodille. *Testit läpäisty.*

2.3 TestPhoneNumber

Yksikkötestit *fi.helsinki.cs.nero.data.PhoneNumber*-luokalle. Testeillä pystyttiin kattamaan kaikki luokan metodit, rivit sekä lauseet.

- **public void testPhoneNumberClone()**
  Testit *PhoneNumber*-luokan konstruktoreille, sekä normaalille että vanhat tiedot kopioivalle versiolle. *Testit läpäisty.*
- **public void testPhoneNumberExceptions()**
  Testejä *PhoneNumber*-luokan konstruktoreille laittomilla arvoilla, joiden tulisi heittää poikkeuksia. *Testit läpäisty.*

- **public void testToString()**
  Testit *toString()*-metodille. *Testit läpäisty.*

- **public void testCompareTo()**
  Testit *compareTo()*-metodille. *Testit läpäisty.*

- **public void testXXX()**
  Testit läpäisty.

2.4 TestPost

Yksikkötestit *fi.helsinki.cs.nero.data.Post*-luokalle. Testeillä pystyttiin peittämään kaikki metodit lukuunottamatta tietokantaa vaativia metodeita, eli työpisteen varauksia käsittelevät metodit sekä työpisteen varaustilanteen kertova *getStatus()*-metodi. Lausekattavuus jäi näin ollen alhaiseksi (40%).

- **public void testPost()**
  Testit *Post*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*

- **public void testGetPostID()**
  Yksikkötestit *getPostID()*-metodille. *Testit läpäisty.*

- **public void testGetRoom()**
  Yksikkötestit *getRoom()*-metodille. *Testit läpäisty.*

- **public void testGetPostNumber()**
  Yksikkötestit *getPostID()*-metodille. *Testit läpäisty.*

- **public void testToString()**
  Yksikkötestit *toString()*-metodille. *Testit läpäisty.*

- **public void testSetPhoneNumbers()**
  Yksikkötestit *setPhoneNumbers()*-metodille. *Testit läpäisty.*

- **public void testGetPhoneNumbers()**
  Yksikkötestit *getPhoneNumbers()*-metodille. *Testit läpäisty.*

2.5 TestProject

Yksikkötestit *fi.helsinki.cs.nero.data.Project*-luokalle. Testeillä pystyttiin kattamaan kaikki luokan metodit, rivit sekä lauseet.

- **public void testProject()**
  Testit *Project*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*

- **public void testGetProjectID()**
  Yksikkötestit *getProjectID()*-metodille. *Testit läpäisty.*
- **public void testGetProjectManager()**
  Yksikkötestit *getProjectManager()*-metodille. *Testit läpäisty.*
- **public void testGetProjectName()**
  Yksikkötestit *getProjectName()*-metodille. *Testit läpäisty.*
- **public void testGetTimeSlice()**
  Yksikkötestit *getProjectID()*-metodille. *Testit läpäisty.*
- **public void testCompareTo()**
  Yksikkötestit *compareTo()*-metodille. *Testit läpäisty.*
- **public void testToString()**
  Yksikkötestit *toString()*-metodille. *Testit läpäisty.*

2.6 TestReservation

Yksikkötestit *fi.helsinki.cs.nero.data.Reservation*-luokalle. Testeillä pystyttiin kattamaan kaikki luokan metodit, rivit sekä lauseet.

- **public void testReservationException()**
  Testit *Reservation*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*
- **public void testClone()**
  Testit *Reservation*-luokan konstruktorille joka kopioi tiedot toiselta Reservation-oliolta. *Testit läpäisty.*
- **public void testGetReservationID()**
  Yksikkötestit *getReservationID()*-metodille. *Testit läpäisty.*
- **public void testGetReservingPerson()**
  Yksikkötestit *getReservingPerson()*-metodille. *Testit läpäisty.*
- **public void testGetTargetPost()**
  Yksikkötestit *getTargetPost()*-metodille. *Testit läpäisty.*
- **public void testGetTimeSlice()**
  Yksikkötestit *getGetTimeSlice()*-metodille. *Testit läpäisty.*
- **public void testGetWeeklyHours()**
  Yksikkötestit *getGetWeeklyHours()*-metodille. *Testit läpäisty.*
- **public void testCompareTo()**
  Yksikkötestit *compareTo()*-metodille. *Testit läpäisty.*

- **public void testToString()**
  Yksikkötestit *toString()*-metodille. *Testit läpäisty.*

2.7 TestRoom

Yksikkötestit *fi.helsinki.cs.nero.data.Room*-luokalle. Testeillä pystyttiin peittämään kaikki metodit lukuunottamatta *getStatus()*-metodia, joka käyttää tietokantaa *Post*-luokan *getStatus()*-metodin kautta. Lausekattavuus kuitenkin 83%.

- **public void testRoom()**
  Testit *Reservation*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*
- **public void testGetRoomID()**
  Yksikkötestit *getRoomID()*-metodille. *Testit läpäisty.*
- **public void testGetBuildingName()**
  Yksikkötestit *getBuildingName()*-metodille. *Testit läpäisty.*
- **public void testGetFloor()**
  Yksikkötestit *getFloor()*-metodille. *Testit läpäisty.*
- **public void testGetRoomNumber()**
  Yksikkötestit *getRoomNumber()*-metodille. *Testit läpäisty.*
- **public void testGetRoomName()**
  Yksikkötestit *getRoomName()*-metodille. *Testit läpäisty.*
- **public void testGetRoomSize()**
  Yksikkötestit *getRoomSize()*-metodille. *Testit läpäisty.*
- **public void testGetRoomDescription()**
  Yksikkötestit *getRoomDescription()*-metodille. *Testit läpäisty.*
- **public void testSetGetPosts()**
  Yksikkötestit *setPosts()*- ja *getPosts()*-metodeille. *Testit läpäisty.*
- **public void testToString()**
  Yksikkötestit *toString()*-metodille. *Testit läpäisty.*

2.8 TestTimeSlice

Yksikkötestit *fi.helsinki.cs.nero.data.TimeSlice*-luokalle. Testeillä pystyttiin kattamaan kaikki luokan metodit, rivit sekä lauseet. *TimeSlice*-luokan määrittely oli kuitenkin liian epäselvä, sillä alku- ja loppupäivämäärien kuuluvuutta aikajaksoon ei oltu mainittu. Tämä antoi mahdollisuuden erilaisille tulkinnoille, jonka syystä ohjelmistosta saattaa löytyä "off-by-one-bugeja.

- **public void testTimeSlice()**
  Testit *TimeSlice*-luokan konstruktorille, sekä laillisilla että virheellisillä arvoilla. *Testit läpäisty.*

- **public void testSetStartDate()**
  Yksikkötestit *setStartDate()*-metodille. *Testit läpäisty.*

- **public void testGetStartDate()**
  Yksikkötestit *getStartDate()*-metodille. *Testit läpäisty.*

- **public void testSetEndDate()**
  Yksikkötestit *setEndDate()*-metodille. *Testit läpäisty.*

- **public void testGetEndDate()**
  Yksikkötestit *getEndDate()*-metodille. *Testit läpäisty.*

- **public void testGetSQLStartDate()**
  Yksikkötestit *getSQLStartDate()*-metodille. *Testit läpäisty.*

- **public void testGetSQLEndDate()**
  Yksikkötestit *getSQLEndDate()*-metodille. *Testit läpäisty.*

- **public void testLength()**
  Yksikkötestit *length()*-metodille. *Testit läpäisty*, mutta epäselvä määrittely alkuperäisessä suunnitteludokumentissa (otetaanko molemmat rajapäivät huomioon vai ei).

- **public void testContains()**
  Yksikkötestit *contains()*-metodille. *Testit läpäisty.*

- **public void testOverLaps()**
  Yksikkötestit *overLaps()*-metodille. *Testit läpäisty.*

- **public void testDaysBetween()**
  Yksikkötestit *daysBetween()*-metodille. Puutteellinen määrittely. *Testit läpäisty.*

- **public void testCommonDays()**
  Yksikkötestit *commonDays()*-metodille. Puutteellinen määrittely. *Testit läpäisty.*

- **public void testStartDayAfter()**
  Yksikkötestit *testStartDayAfter()*-metodille. *Testi epäonnistui*. Syynä taas virheellinen määrittely. Lisäksi metodin nimi on hämäävä. Vika on tällä kertaa itse testissä (yhden päivän heitto).

- **public void equals()**
  Yksikkötestit *equals()*-metodille. *Testit läpäisty.*

- **public void testCompareTo()**
  Yksikkötestit *compareTo()*-metodille. *Testit läpäisty.*

- **public void testToString()**
  Yksikkötestit *toString()*-metodille. *Testit läpäisty.*

2.9 TestSession

Yksikkötestit *fi.helsinki.cs.nero.logic.Session*-luokalle. Koska *Session*-luokan metodeista suurin välittää tietonsa *NeroDatabase*-tietokantaluokalle, on metodeista testattu vain 56% (28/50). Tästä johtuen myös lausekattavuus jäi 41%:iin. Todettakoot, että kaikille testatuille metodeille saatiin 100% lausekattavuus.

- **public void testFilterTimescale()**
  Yksikkötestit *setFilterTimescale()*- ja *getFilterTimeScale*-metodeille. *Testit läpäisty.*
- **public void testFilterProject()**
  Yksikkötestit *setFilterProject()*- ja *setFilterProject*-metodeille. *Testit läpäisty.*
- **public void setFilterPersonName()**
  Yksikkötestit *setFilterPersonName()*- ja *getFilterPersonName*-metodeille.
  textitTestit läpäisty.
- **public void testFilterRoomName()**
  Yksikkötestit *setFilterRoomName()*- ja *getFilterRoomName*-metodeille. *Testit läpäisty.*
- **public void testFilterEndingContracts()**
  Yksikkötestit *setFilterEndingContracts()*- ja *getFilterEndingContracts*-metodeille. *Testit läpäisty.*
- **public void testFilterMaxPosts()**
  Yksikkötestit *setFilterMaxPosts()*- ja *getFilterMaxPosts*-metodeille. *Testit läpäisty.*
- **public void testFilterWithoutPost()**
  Yksikkötestit *setFilterWithoutPost()*- ja *getFilterWithoutPost*-metodeille. *Testit läpäisty.*
- **public void testFilterFreePosts()**
  Yksikkötestit *setFilterFreePosts()*- ja *getFilterFreePosts*-metodeille. *Testit läpäisty.*
- **public void testFilterPartTimeTeachers()**
  Yksikkötestit *setFilterPartTimeTeachers()*- ja *getFilterPartTimeTeachers*-metodeille. *Testit läpäisty.*
- **public void testTimeScaleSlice()**
  Yksikkötestit *setTimeScaleSlice()*- ja *getTimeScaleSlice*-metodeille. *Testit läpäisty.*
- **public void testActiveRoom()**
  Yksikkötestit *setActiveRoom()*- ja *getActiveRoom*-metodeille. *Testit läpäisty.*

- **public void testRegisterObserver()**
  Yksikkötestit *registerObserver()*-metodille. *Testit läpäisty.*

- **public void testStatusMessage()**
  Yksikkötestit *setStatusMessage()*- ja *getStatusMessage*-metodeille. *Testit läpäisty.*

- **public void testWaitState()**
  Yksikkötestit *waitState()*-metodille. *Testit läpäisty.*

2.10 TestNeroObserverManager

Yksikkötestit tapahtumankäsittelyluokalle *fi.helsinki.cs.nero.event.NeroObserverManager*. Aidosti kattavat testit yksinkertaiselle luokalle. Vähintään 101% lausekattavuus. Toteuttaa myös itse NeroObserver-rajapinnan.

- **public void testAddObserver()**
  Yksikkötestit *addObserver()*-metodille. *Testit läpäisty.*

- **public void testNotifyObservers()**
  Yksikkötestit *notifyObservers()*-metodille. *Testit läpäisty.*

3 Järjestelmätestaus

Järjestelmätestauksessa käytiin läpi suunnitteludokumentissa[Ner04b] kuvatut esimerkkikäyttötapaukset, jotka kattoivat vaatimusdokumentissa[Ner04a] kuvatut ohjelmistolle asetetut toiminnalliset vaatimukset. Kaikki käyttötapaukset onnistuttiin viemään läpi kattaen lähes kaikki toiminnalliset vaatimukset.

Myös järjestelmätestauksen yhteydessä käytettiin EMMA-työkalua lausekattavuuden mittaamiseksi. Käyttötapausten läpiviennin avulla saavutettiin 89% lausekattavuus, joka käsittää lähes kaiken koodin lukuunottamatta poikkeusten- ja muiden virhetilanteiden käsittelyä. Voidaan siis todeta, että käyttötapaukset kattoivat koko koodin, ja koodi puolestaan toteutti kaikki määritellyt esimerkkikäyttötapaukset. Taulukossa 2 on esitelty lausekattavuudet pakkauksittain. Järjestelmätestauksen jälkeen *fi.helsinki.cs.nero.ui*-pakkauksesta poistettiin muutamia turhia käyttöliittymätestailussa käytettyjä luokkia. Tästä johtuen siihen liittyvät kattavuusprosentit ovat todellisuudessa hieman korkeampia.

3.1 Käyttötapaukset

- **KT1 - Kaksi uutta projektityöntekijää**
  Kummatkin työntekijät löytyvät "ilman työpistettä"haulla. Hakemalla kartasta projektin työhuoneita ja kahta vapaata työpistettä uudet työntekijät saadaan sijoitettua lähelle projektin muita työskentelijöitä.

Luokka	Metodikattavuus		Lausekattavuus		Rivikattavuus	
fi.helsinki.cs.nero	3/4	75%	57/86	66%	12.8/21	61%
fi.helsinki.cs.nero.data	71/84	85%	954/1177	81%	251.9/301	84%
fi.helsinki.cs.nero.db	24/26	92%	1878/2109	89%	383.7/431	89%
fi.helsinki.cs.nero.event	3/3	100%	60/68	88%	15/17	89%
fi.helsinki.cs.nero.logic	43/50	86%	704/874	81%	182/220	83%
fi.helsinki.cs.nero.ui	169/205	82%	7733/8439	92%	1520.7/1679	91%
Yhteensä	313/372	84%	11386/12753	89%	2366.1/2669	89%

Taulukko 2: Järjestelmätestauksessa saavutetut lausekattavuudet pakkauksittain

- **KT2 - Työpisteen historia**
  Asettamalla tarkasteltava aikaväli alkamaan ja päättymään sopivasti saadaan selville huoneen silloinen varaustilanne. Hakua voidaan nopeuttaa karsimalla oikeassa reunassa näytettävien ihmisten määrää hakukenttien avulla.

- **KT3 - Uusi sivutoiminen**
  Rajaamalla haku sivutoimisiin saadaan selville sivutoimisille tuntiopettajille tarkoitettu huone. Kyseinen huone on kuitenkin luultavasti jo ohjelman käyttäjän tiedossa, joten hakua voidaan rajata lisää valitsemalla myös "ilman työpistettä-hakuehto. Kun sivutoimisten työpiste on valittu kartasta, voidaan henkilö joko sijoittaa jonkun toisen sivutoimisen kanssa samaan työpisteeseen tai vaihtoehtoisesti voidaan luoda uusi työpiste, johon uusi työntekijä sitten sijoitetaan.

- **KT4 - Työpisteen poistaminen**
  Työpiste voidaan poistaa raahaamalla se käyttöliittymässä roskakoriin. *Työpistettä ei voi poistaa jos siihen liittyy yksikin varaus tai puhelinnumero.*

- **KT5 - Päättyvät työsuhteet**
  Ohjelmassa on hakuehdot "näytä päättyvät sopimukset", joka näyttää kyseiset tiedot asetetulta aikaväliltä. Sopivat hakuehdot ovat vakiona valittuina ohjelman käynnistyksen jälkeen, aikavälin ollessa kolme kuukautta nykyisestä päivämäärästä eteenpäin.

- **KT6 - Työhuoneen tiedot**
  Huoneen tiedot saa näkyviin joko valitsemalla se pohjapiirroksesta tai oikealla hiiren painikkeella henkilölistassa näkyvästä työpistevarauksesta.

- **KT7 - Henkilön tiedot**
  Henkilöä voidaan hakea siihen tarkoitetulla hakukentällä. Hakua voidaan tietysti myös rajata. Näkyviin tulee henkilön varaus- ja sopimustiedot valitulta aikaväliltä.

- **KT8 - Projekti 4**
  Tiettyyn projektiin kuuluvat työhuoneet saadaan selville valitsemalla projektilistasta sopiva projekti, jolloin henkilölistaa rajataan näyttämään vain kyseiseen projektiin kuuluvat henkilöt, sekä kartalta kehystetään työhuoneet joiden työpisteissä on

projektiin kuuluvien henkilöiden varauksia. *Tällä hetkellä projektilistassa on suuri määrä ns. yhden hengen projekteja, ja valintaa voi olla vaikea selata.*

- **KT9 - Puhelinnumerot**
  Työpisteisiin liittyviä puhelinnumeroita voidaan hallinnoida siihen erikseen avattavalla dialogilla, joka saadaan näkyviin klikkaamalla työpistetiedoissa näkyvää puhelinnumeroikonia. Puhelinnumeroita voidaan liittää työpisteisiin, jolloin se siirretään mahdollisesta edellisestä työpisteestä. Samaan työpisteeseen voi kuulua useampi eri puhelinnumero.

3.2 Huomiot vaatimuksiin

- **Laadullinen vaatimus VL3 - Uudelleenkäytettävyys**
  Joistakin ohjelman osista saattoi vahingossa tulla uudelleenkäytettäviä... ;>

- **Laadullinen vaatimus VL6 - Suorituskyky**
  Ohjelma lataa tietokannasta suuren määrän tietoja uusiksi jokaisen päivityksen yhteydessä, jotta voidaan varmistua tietojen oikeellisuudesta. Joissain tilanteissa odotusajat saattavat kasvaa.

- **Laadullinen vaatimus VL7 - Käyttöympäristö**
  "Ohjelman täytyy toimia laitoksen verkosta, muttei sen ulkopuolelta."Oikeasti vaatimuksella tarkoitettiin, että olisi hienoa jos ohjelmaa voisi käyttää myös muualtakin. Tällöin tietokantahakujen ajat kasvavat, mutta eivät sietämättömiksi. Ohjelmistoa onkin pääasiassa kehitetty laitoksen ulkopuolelta.

- **Toiminnallinen vaatimus VT15 - Henkilön haku**
  Haettaessa henkilöä tulee hänellä olla joko voimassa oleva sopimus tai työpistevaraus joka leikkaa asetettua aikaväliä.

- **Toiminnallinen vaatimus VT18 - Henkilökunnan puhelinluettelo**
  Ohjelmisto ei tarjoa henkilökunnan puhelinluetteloa, vaan pelkästään työpisteisiin kuuluvat puhelinnumerot.

- **Toiminnallinen vaatimus VT21 - Huoneen käyttötarkoituksen muuttaminen**
  Työhuoneen pitää olla lisättynä tietokantaan, jotta siihen voi liittää työpisteitä. Lisäksi työhuoneen tulee löytyä SVG-muotoisest kartasta.

- **Toiminnallinen vaatimus VT29 - Rooma korjaus 2**
  Rooman korjaamisesta luovuttiin suunnitteluvaiheessa, koska sen todettiin syövän liikaa resursseja, eikä Rooman koodia tultaisi uudelleenkäyttämään.

4 Löydetyt virheet

Allaolevassa listauksessa projektin toteutusvaiheessa sekä järjestelmätestauksessa löydetyt virheet. Suurin osa virheistä on korjattu, eikä fataaleja ongelmatapauksia pitäisi esiin-

työ. Osa ongelmista liittyy ulkopuolisiin tekijöihin, kuten käytettyihin kirjastoihin ja muuhun ympäristöön.

Virhe N1 **Kartta ja Batik-kirjasto**
Toteutusvaiheen loppupuolella havaittiin, että huoneiden vaihto varaa muistia. Tarkempi tutkiskelu osoitti, että vika on Batik-kirjastossa (SVG-renderöijä). Vika korjattiin osittain välttämällä turhaa kerrosten vaihtoa huoneen vaihtumisen yhteydessä. Kyseinen kirjasto kuitenkin yhä varaa muistia, ja ohjelma saattaa kaatua jos kerrosta vaihdetaan kymmeniä kertoja. Tämän ei pitäisi kuitenkaan tapahtua normaalissa käytössä.
Ratkaisu: Ohjelma on hyvä sammuttaa jos sen käyttöä ei ole heti jatkamassa. Tulevaisuudessa uudemmat Batik-kirjastot saattavat myös auttaa asiaa.
Tila: avoin

Virhe N2 **Henkilön nimen/projektin nimen asettelu**
Jos sopimusjakso tai varausjakso alkoi ennen asetettua aikaväliä, eivät kyseisen jakson tiedot näkyneet jaksoa kuvaavassa palkissa kokonaan.
Ratkaisu: Kyseisten JPaneleiden layout managereiksi asetettiin SpringLayout, jolla tekstit saatiin sijoitettua näkyviin.
Tila: korjattu

Virhe N3 **Aikavälin resetointi**
Järjestelmätestauksen yhteydessä todettiin, että olisi hyödyllistä voida asettaa aikaväli takaisin oletukseksi.
Ratkaisu: Aikavälikenttien viereen lisättiin nappi resetoimista varten. Tämä vaati pieniä muutoksia *Session*-luokan toteutukseen.
Tila: korjattu

Virhe N4 **Henkilöitä näkyy tuplana**
Henkilölistauksessa jotkin henkilöt näkyvät kahteen kertaan.
Ratkaisu: Todettiin, että varsinainen vika on tietokannassa olevissa tiedoissa. Joillekin henkilöille oli jostain syystä luotu henkilötunnisteet kahteen kertaan, jolloin kannassa esiintyi kaksi eri henkilöksi luokiteltavaa täsmälleen saman nimistä henkilöä. Tämän korjaamiseksi tietokanta tulisi käydä läpi duplikaattien osalta.
Tila: avoin

Virhe N5 **Varausta on mahdollista venyttää ikkunan ulkopuolelle**
Muutettaessa työpistevarauksen kokoa sitä voidaan venyttää yli asetetun aikavälin ikkunan ulkopuolelle. Ohjelmistoa kuitenkin käytetään useimmiten koko näytön vievässä tilassa, eikä kyseistä ominaisuutta nähty tarpeelliseksi rajoittaa.
Ratkaisu: -
Tila: todettu

Virhe N6 **Huonepaneeli ei päivity tietoja muutettaessa**
Todettiin, ettei huonepaneelin tiedot muutu siihen liittyviä puhelinnumeroita tai varaustietoja muutettaessa.

Ratkaisu: Huonepaneelin kuuntelijasta puuttui kyseisesten muutosten käsittely. Nyt huonepaneeli kuuntelee oikeita tapahtumatyyppejä ja päivittää tiedot tarvittaessa.
Tila: korjattu

Virhe N7 **Saman työpisteen varaukset piirtyvät päällekkäin**
Huomattiin, että jos samaan työpisteeseen loi eri mittaisia mutta osittain päällekkäin meneviä varauksia, ne saattoivat piirtyä samalle riville. Päällekkäiset varaukset samaan työpisteeseen ovat harvinaisia, mutta mahdollisia, etenkin sivutoimisten tuntiopettajien tapauksessa.
Ratkaisu: Kyseisten jaksojen vertailussa oli virhe. Nyt luodaan tarvittaessa uusia rivejä. Sama ongelma esiintyi myös jos henkilöllä oli päällekkäisiä työsopimusjaksoja.
Tila: korjattu

Virhe N8 **Poikkeus käytettäessä Java 1.5.0:aa ja X-ikkunointijärjestelmää**
Toteutusvaiheen loppupäivinä laitokselle asennettiin Javan versio 1.5.0. Huomattiin, ettei allaolevaa virheilmoitusta saatu aiemmilla Javan versioilla. Virhe ei vaikuta ohjelmiston suoritukseen millään tapaa. Virheviesti ilmaantuu satunnaisesti ikkunaa luodessa, siirrettäessä tai suljettaessa. Virheestä kertova stack trace on taulussa 3.
Ratkaisu: -
Tila: todettu

```
Atom was 0
Exception on Toolkit thread: java.lang.NullPointerException:
    Failed to retrieve atom name.
java.lang.NullPointerException: Failed to retrieve atom name.
    at sun.awt.X11.XlibWrapper.XGetAtomName(Native Method)
    at sun.awt.X11.XAtom.<init>(XAtom.java:219)
    at sun.awt.X11.XAtom.get(XAtom.java:146)
    at sun.awt.X11.XAtom.getAtomListProperty(XAtom.java:630)
    at sun.awt.X11.XAtom.getAtomListProperty(XAtom.java:650)
    at sun.awt.X11.XNETProtocol.getState(XNETProtocol.java:109)
    at sun.awt.X11.XWM.getExtendedState(XWM.java:1024)
    at sun.awt.X11.XWM.isStateChange(XWM.java:1075)
    at sun.awt.X11.XFramePeer.handlePropertyNotify(XFramePeer.java:343)
    at sun.awt.X11.XBaseWindow.dispatchEvent(XBaseWindow.java:822)
    at sun.awt.X11.XWindowPeer.dispatchEvent(XWindowPeer.java:437)
    at sun.awt.X11.XBaseWindow.dispatchToWindow(XBaseWindow.java:766)
    at sun.awt.X11.XToolkit.dispatchEvent(XToolkit.java:394)
    at sun.awt.X11.XToolkit.run(XToolkit.java:493)
    at sun.awt.X11.XToolkit.run(XToolkit.java:438)
    at java.lang.Thread.run(Thread.java:595)
```

Taulukko 3: Satunnainen X-ikkunointijärjestelmään liittyvä virhe ajettaessa ohjelmistoa Java 1.5.0:lla

Lähteet

Ner04b Kuusela, J., Muukkonen, J., Sjöblom, T., Sundberg, V., Suominen, O. ja Tuohenmaa, T., *Ohjelmistotuotantoprojekti Nero, Suunnitteludokumentti*. Helsingin yliopiston tietojenkäsittelytieteen laitos, 2004.

Ner04c Kuusela, J., Muukkonen, J., Sjöblom, T., Sundberg, V., Suominen, O. ja Tuohenmaa, T., *Ohjelmistotuotantoprojekti Nero, Testaussuunnitelma*. Helsingin yliopiston tietojenkäsittelytieteen laitos, 2004.

Ner04a Kuusela, J., Muukkonen, J., Sjöblom, T., Sundberg, V., Suominen, O. ja Tuohenmaa, T., *Ohjelmistotuotantoprojekti Nero, Vaatimusdokumentti*. Helsingin yliopiston tietojenkäsittelytieteen laitos, 2004.