

Ilari Kajaste and Timo Poranen (eds.)

Software Projects 2007-2008



DEPARTMENT OF COMPUTER SCIENCES
UNIVERSITY OF TAMPERE

D-2008-8

TAMPERE 2008

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER SCIENCES
SERIES OF PUBLICATIONS D – NET PUBLICATIONS
D-2008-8, SEPTEMBER 2008

Ilari Kajaste and Timo Poranen (eds.)

Software Projects 2007-2008

DEPARTMENT OF COMPUTER SCIENCES
FIN-33014 UNIVERSITY OF TAMPERE

ISBN 978-951-44-7473-6
ISSN 1795-4274

Preface

This report contains project stories of 14 software development projects and one usability team. The students to the projects came from the Project Work course or from the Software Project Management course. The stories describe how the project went and what kind experiences the groups got during their project. In the end of each story there are statistics of the project.

Table 1 contains general course statistics (number of projects and usability teams, number of students in the courses and average project size in working hours) starting from academic year 2002. Some numbers from earlier courses are unknown; these are marked with a question mark (?).

Table 1: Course statistics 2002-2008.

Academic year	Projects	Usability teams	PW students	SPM students	Average project size
2002-3	15	0	?	?	?
2003-4	15	0	?	?	?
2004-5	13	0	60	7	?
2005-6	19	1	98	8	1008 hours
2006-7	18	2	87	34	1089 hours
2007-8	14	1	70	29	997 hours

Table 2 gives an overview of the projects of this year. The table lists project's name, project's type (WWW = a WWW application often containing a database, Appl. = a stand-alone application), client (Dept. = Department of Computer Sciences, Univ. = other university units, Assoc. = association, Company = a commercial company), or used software development model (WF = Waterfall, INC = Incremental, ITER = Iterative or Agile = Agile development model, not any specific but more like a model where selected features from different agile development models are used), group size (Number of managers + size of the project team + usability experts), and total working hours of the project.

During the course project managers kept up the working hours of the project team. The working hours are divided into nine activities: project planning and management, requirements specification, software design, code (programming), integration and testing, reviews, repair (bug fixing), studying and other. All projects, except usability team, gave their working hours divided into these categories.

The statistic sections of almost all projects contain tables for general

Table 2: General project statistics.

Project	Type	Client	Dev. Model	Group	Hours
Uteam	-	Dept.	All	2+11	2128
Hirttosilmukka	WWW	Assoc.	WF	2+4+1	830
HAT	WWW	Dept.	ITER	2+3+1	833
WUH 2.0	Appl.	Assoc.	INC	2+4+1	1200
Aateliset	WWW	Assoc.	Scrum	2+5+1	1222
Tilatar	WWW	Assoc.	INC/WF	1+4+2	1015
Janis	Appl.	Dept.	Agile	2+5+2	1240
Users	WWW	Dept.	INC	2+4+1	600
Statua	Appl.	Company	Scrum	2+5+1	939
Scifleffa	WWW	Company	INC	2+4	1008
AgileTool	WWW	Company	Scrum	2+5+2	1090
Reap	Appl.	Company	INC	1+5+1	1330
Gaze Games	Appl.	Dept.	INC	3+5+2	1086
RVLE	Appl.	Dept.	ITER	2+3	708
EDP2008	WWW	Univ.	INCR	2+3+2	853

project information, requirements and high-level design outcome, design outcome, projects documents, codelines, and productivity metrics.

This was the third year that the course had a separate usability team to make sure usability concerns are taken care of in each project, and to provide coordinated help with them. While organizing the team's interaction with projects still needs more work, the concept has proven useful and will gladly be continued, and improved upon, next year.

We thank our clients for their interesting project topic suggestions and quest lecturers of the course, Tero Ahtee (Tampere University of Technology, Department of Software Systems) and Tero Lahtinen (Nokia Siemens Networks). And finally, special thanks goes to all project work and software project management course students, you did it again!

Tampere, October 2008
Ilari Kajaste and Timo Poranen

Contents

1 Käytettävyyssryhmä	1
1.1 Yleistä	1
1.2 Organisaatio ja johtaminen	1
1.3 Metodit ja työkalut	3
1.4 Yhteenvedo	3
1.5 Tilastot	4
2 Hirttosilmukka	6
2.1 Overview	6
2.2 Organisation and management	6
2.3 Methods and tools	7
2.4 Project phases and development model	7
2.5 Conclusions	7
2.6 Screenshots	8
2.7 Statistics	9
3 HAT 2007	12
3.1 Overview	12
3.2 Organisation and management	12
3.3 Methods and tools	13
3.4 Project phases and development model	13
3.5 Conclusions	13
3.6 Sample screenshot	14
3.7 Statistics	14
4 WUH 2.0	16
4.1 Yleistä	16
4.2 Organisaation ja johtaminen	16
4.3 Menetelmät ja työkalut	17
4.4 Projektin vaiheet ja kehitysmalli	18
4.5 Yhteenvedo	18
4.6 Statistics	18
5 Aateliset	22
5.1 Johdanto	22
5.2 Organisaatio	22
5.3 Menetelmät ja käytetyt ohjelmistot	23
5.4 Projektin eteneminen	24
5.5 Johtopäätökset	26

5.6	Tilastot	26
6	TILATAR	30
6.1	Yleistä	30
6.2	Projektiorganisaatio	31
6.3	Metodit ja työkalut	32
6.4	Projektin vaiheet	33
6.5	Johtopäätökset	34
6.6	Tilastoja	34
7	Janis	37
7.1	Yleiskuvaus projektista	37
7.2	Organisaatio ja projektin hallinta	38
7.3	Menetelmät ja työkalut	39
7.4	Projektin vaiheet ja kehitysmalli	40
7.5	Johtopäätökset	41
7.6	Statistiikkaa	42
8	Laitteistotietokanta	45
8.1	Yleistä	45
8.2	Projektiorganisaatio	45
8.3	Käytetyt välineet ja työkalut	46
8.4	Projektin vaiheet ja kehitysmalli	46
8.5	Johtopäätökset	47
8.6	Tilastot	47
9	Statua	49
9.1	Overview	49
9.2	Organisation and management	49
9.3	Methods and tools	49
9.4	Sample screenshots	50
9.5	Homepage	51
9.6	Project phases and development model	51
9.7	Conclusions	51
9.8	Statistics	51
9.9	Codelines	53
10	Scifi	54
10.1	Yleistä	54
10.2	Projektiryhmän jäsenet	54
10.3	Käytetyt välineet	55

10.4	Projektin vaiheet	56
10.5	Johtopäätökset	56
10.6	Tilastoja	56
11	AgileTool	59
11.1	Yleiskuvaus projektista	59
11.1.1	Tuote ja ympäristö	59
11.1.2	Ohjelman toimiminen osana järjestelmää	60
11.1.3	Rajallisuudet ja rajoitteet	60
11.2	Projektioorganisaatio	60
11.3	Työskentelymetodit ja työkalut	61
11.4	Projektin vaiheet	62
11.5	Yhteenvedo	62
11.6	Tilastoja	64
12	ReaP	66
12.1	Overview	66
12.2	Organisation and management	66
12.3	Methods and tools	66
12.4	Project phases and development model	68
12.5	Conclusions	68
12.6	Statistics	69
13	Gaze Games	72
13.1	Overview	72
13.2	Organisation and management	72
13.3	Methods and tools	73
13.4	Conclusions	74
13.5	Statistics	74
14	RVLE	77
14.1	Overview	77
14.2	Organisation and management	78
14.3	Methods and tools	78
14.4	Project phases and development model	80
14.5	Conclusions	81
14.6	Acknowledgement	81
14.7	Statistics	81

15 EDP2008	84
15.1 Overview	84
15.2 Organisation and management	85
15.3 Methods and tools	86
15.4 Project phases and development model	87
15.5 Conclusions	87
15.6 Statistics	89

1 Käytettävyysryhmä

1.1 Yleistä

Tämän vuoden käytettävyystiimi (Uteam) perustuu kahden viime vuoden pohjalle. Uteam oli toiminnassa ensimmäisen kerran siis lukuvuonna 2005-2006. Uteamin projektiryhmä koostui tänä vuonna 11 jäsenestä, josta yksi jatkoi viime vuodelta. Lisäksi ryhmässä toimi tänä vuonna kaksi projektipäällikköä.

Uteamin idea on, että ryhmän jäsenet osallistuvat muihin projekteihin käytettävyyteen erikoistuneina henkilöinä ja expertteinä. Uteam pitää huolta, että jokaisen projektin käytettävyystarpeet on tyydytetty. Uteamin jäsenet jakautuivat projekteihin aluksi sen mukaan kuin käytettävyystarpeita ajateltiin olevan. Tänä lukuvuonna oli 14 projektia ja melkein jokainen ryhmäläinen työskenteli vähintään kahdessa projektissa.

1.2 Organisaatio ja johtaminen



Figure 1: Projektipäälliköt Juha-Pekka Honkavaara ja Juha Mattila.

Uteamin henkilöstöön kuuluivat: Arttu Ekholm, Johanna Huhtala, Matti Huokko, Saira Koiranen, Jussi Maaniitty, Santtu Mansikkamaa, Marko Oravainen, Juha Pakkanen, Jenni Päckilä ja Hanna Venesvirta. Lisäksi Uteamiin kuului Riki Kawakami joka oli mukana vain syyslukukaudella. Projektipäälliköinä toimivat Juha-Pekka Honkavaara ja Juha Mattila. (Kuvat puuttuvat seuraavista henkilöistä: Marko Oravainen, Jenni Päckilä, Santtu Mansikkamaa, Riki Kawakami)

Uteamiläiset keskittyivät projekteissa työskentelemiseen ja, koska niissä on omat tapaamisensa päätimme pitää tapaamisia Uteamin sisällä epäsäännöllisin väliajoin. Päätimme tavata silloin kun oli kerääntynyt käsiteltäviä asioita

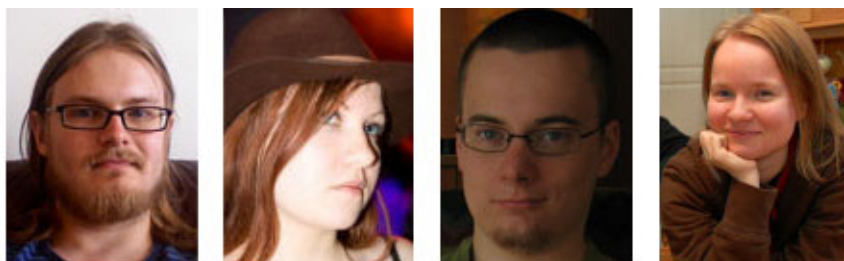


Figure 2: Projektiryhmäläiset Arttu Ekholm, Johanna Huhtala, Matti Huokko ja Saira Koiranen.



Figure 3: Projektiryhmäläiset Jussi Maaniitty, Juha Pakkanen ja Hanna Venesvirta.

tai jos pidimme arviointitilaisuuksia. Tapaamisia ryhmän kesken oli yhteensä 11. Näissä tapaamisissa käsiteltiin ryhmäläisten edistymistä projekteissaan, mahdollisia ongelmia, vertaisarviointeja, katselmointeja ja tarvittavien dokumenttien kirjoittamista. Lisäksi kahdessa tapaamisessa viime vuoden projektipäällikkö luennoi käytettävistä metodeista.

Uteam ei varsinaisesti tuottanut itselleen paljon dokumentteja projektisuunnitelman ja loppuraportin lisäksi vaan dokumentit tuotettiin muihin projekteihin. Omien projektien lisäksi Uteamin jäsenet menivät avuksi muihin projekteihin jos sitä tarvittiin tai menttiin avuksi pelkästään esim. testaukseen. Yksi Uteamin hyvistä puolista oli ryhmäläisten apu toisilleen käytettävyyssasioissa.

Käytössämme oli viime vuoden materiaali, joista oli paljon apua. Viime vuoden ryhmä oli kirjoittanut paljon dokumenttipohjia ja ohjeita valmiiksi, jotka pystyttiin suoraan ottamaan käyttöön tämän vuoden ryhmässä. Tämä vähensi huomattavasti projektipäälliköiden työtaakkaa, mikä oli tarkoituksenakin. Viime vuoden ryhmältä jäi myös kotisivupohja jota pystyttiin käyttämään Uteamin tiedotuskanavana, dokumenttien jakeluun ja tuntikirjanpitoon. Nämä kaikki jäävät myös seuraavan käytettävyystiimin käyttöön. Uteamin kotisivut löytyvät osoitteesta: <http://www.cs.uta.fi/~uteam/>.

1.3 Metodit ja työkalut

Uteam käytti seuraavia työkaluja ja metodeja projektin aikana:

- Heuristinen arviointi (7)
- Vertaisarviointi (3)
- Käytettävyystestaus (6)
- Paperi prototyyppi testaus (3)
- Käyttöliittymä haastattelut (1)

Näistä vertaisarviointi oli hyödyllisin. Vertaisarvioinnin ideana on, että joku esittää käyttöliittymäsuunnitelmansa ja sitten ryhmän sisällä arvioidaan kyseistä käyttöliittymää. Tämä auttaa kyseisen käyttöliittymän suunnittelijaa näkemään laajemmin mikä on toimivaa ja mikä ei, koska saadaan suurempi näkökulma asiaan. Pääasiana pitäisi olla parannusehdotusten esittäminen, ei pelkästään virheiden osoittaminen.

Toinen hyödyllinen metodi oli tietenkin käytettävyystestaus millä tahansa käyttöliittymällä. Käytettävyystestaus osoittaa miten normaali ensikerran käyttäjä osaa käyttää käyttöliittymää.

Uteam tuotti käytettävyykdokumentteja kurssin aikana seuraavasti:

- Käytettävyysskartoitus (11, 142 pages)
- Käyttöliittymäsuunnitelma (9, 303 pages)
- Testaussuunnitelma (3, 39 pages)
- Testausraportti (3, 51 pages)

Näistä käytettävyysskartoitusta ja käyttöliittymäsuunnitelmaa tarvittiin eniten projekteissa ja niihin panostettiin myös eniten.

1.4 Yhteenveto

Suurimpana haasteena käytettävyystiimin työskentelyssä oli oman projektin koordinointi muiden projektien kanssa. Oli hieman vaikeuksia pysyä mukana, mitä mikäkin projekti teki ja missä vaiheessa se oli. Myös kommunikointi muiden projektipäälliköiden kanssa oli vähäistä ja hankalاهkoa. Heidän oli vaikea tietää, mikä käytettävyystiimin tehtävä loppujen lopuksi on ja miten he voivat hyödyntää käytettävyyshenkilöitä. Tämä siitä huolimatta,

että lähetimme projektipäälliköille dokumentin, jossa kuvattiin käytettävyyssiimin toimintaa. Ongelmista huolimatta käytettävyyssiimi oli hyödyllinen resurssi projekteille. Käytettävyyssiimin jäsenet työskentelivät ahkerasti eri projekteissa. Käytettävyyssiimin jäsenet kokivat osallistumisensa projektikurssille hyväksi asiaksi. Kaikenkaikkiaan käytettävyyssiimi onnistui perustehtävässään hyvin.

1.5 Tilastot

Käytettävyyssiimi työskenteli 2128,25 tuntia tänä lukuvuonna. Tämä luku ilman ryhmän kahta projektipäällikköä on 1832,25. Poiketen muista projektikurssin ryhmistä, käytettävyyssiimillä oli oma tuntijaottelunsa käytössä. Siinä tunnit jaettiin seuraaviin ryhmiin: Meeting, Testing, Study ja Other. Eli tapaamiset, testaamiset, asioiden opettelu ja viimeisenä muut projektityöskentelyyn liittyvät toimenpiteet. Seuraavasta graafista ilmenee nämä jakaumat.

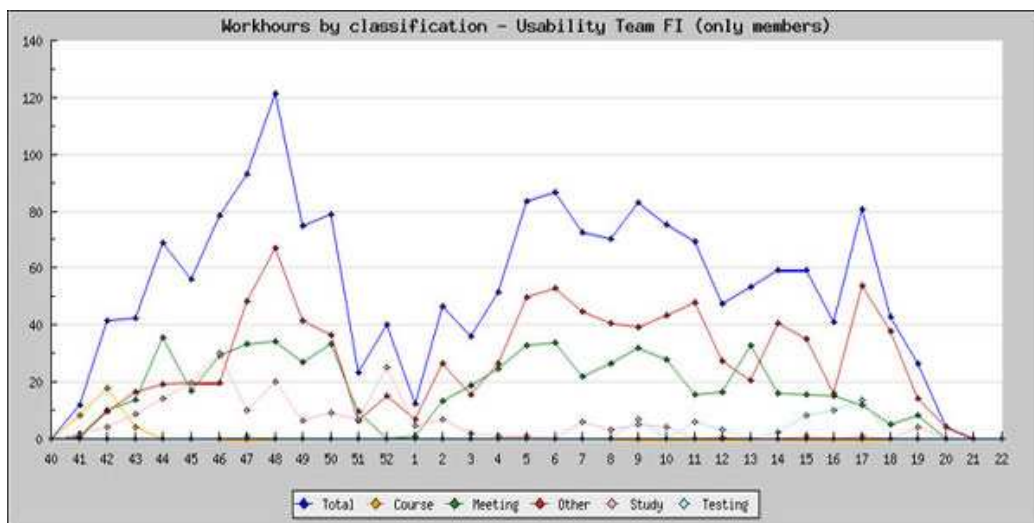


Figure 4: Työtuntien jakaumat.

Käytettävyyssiimin jäsenten tehtävänä oli toimia heille nimetyissä projekteissa, eikä niinkään käytettävyyssiimissä. Tämä graafi kuvaa sitä.

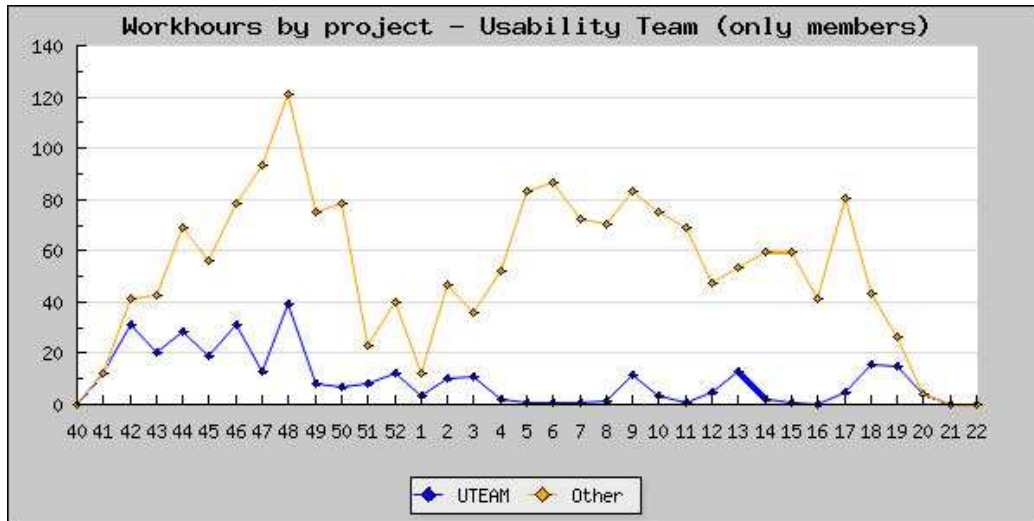


Figure 5: Työtunnit Uteam ja muut.

Table 3: Yleistietoa projektista.

Ryhmä- koko	Keh. malli	Aloitus pvm	Lopetus pvm	Päivät	Tunnit	Tunnit / (Päivät*Ryhmä)
2+11	N/A	12.10.2007	15.5.2008	216	2128,25	0.9

Table 4: Dokumentaatio.

Dokumentti	Sivuja	Versioita
Esitutkimus	7	2
Projektisuunnitelma	38	1
Käytettävyysuunnitelmat	303	9
Käytettävyystestiraportit	51	3
Loppuraportti	41	4
Loppukertomus	5	1
Viikkoraportit	28	
Katselmointiraportit	3	
Yhteensä	476	

2 Hirttosilmukka

2.1 Overview

This document shortly describes the Hirttosilmukka-team work for the Project Course 2008. Project subject was to create from scratch a new web portal for Luuppi Ry (customer), containing at least content management, membership registry and event management features. From these, only the first two features were achieved. However, project development continues during summer in separate, non-course related project sponsored and endorsed by the customer. Software screenshots can be found at the end of the document.

2.2 Organisation and management



Figure 6: Project members.

Project personnel included two project managers and five project staff members. This reduced into four by November, since project member left the project due time issues.

Project managers were Aki Tuomi and Tuomas Tauriala. Project staff was Juuso Mäkinen, Pekka Jääskeläinen, Lauri Renko ja Visa-Valtteri Pimiä. Usability group member Arttu Ekholm was assigned to our group as usability expert.

Project communication tools were mailing list, IRC and telephone. Project team met once a week at university premises for a weekly status, design and planning meeting. These meetings were held to monitor project status, to assign new tasks and to do some collaborative work for the project, such as document finalizing and inspection.

Since project had two managers, it was immediately clear that a task division was required. Project managers agreed to split tasks so that Tuomi handles production issues, and Tauriala handles human resources, reporting and other miscellaneous issues. This split was quite working one, and it

removed the risk of decision collisions and redundant work. It also enabled the project managers to fully focus on their given task.

2.3 Methods and tools

The product was built on MVC (model-view-controller) compatible framework, so design was done in MVC style. The framework used was CakePHP, which is relatively new system. Views were done using Smarty template language. Programming language used here was PHP.

For development we used Eclipse development environment. Software version control was done using Subversion system. The software was developed using a development environment which was ran elsewhere, completely independent of the actual production system.

We encountered some challenges with CakePHP, namely the problem that it requires one to have an dummy application directory, so instead of having `http://www.luuppi.fi/` we are forced by design to use `http://www.luuppi.fi/luuppi`. The amount of work required to bypass this was considered to be too heavy , so it was left alone.

2.4 Project phases and development model

The project followed the waterfall model, and was phased as such. Project definition and design phase lasted the entire fall episode, and was fully ended by January. Production phase started fully at February, due to document approval and fixing. Testing and integration phase was not reached, and only some of the testing was done.

The planning and design phase produced product specifications, implementation and testing plan, usability plan and usability testing plan. From project hours, this phase used most of the hours.

The project problems started at the implementation phase. Project members had too little time to complete their assignments, so the project tasks started to accumulate. By March, this problem was evident and decision was made to change the scope of the project to allow partial completion.

Testing phase consisted only of testing the completed features, content management and membership registry.

2.5 Conclusions

From the various metrics, it is fairly simple to see that the main focus on this project has been on documentation. Lines of code produced in one project month is roughly half less than assumed lines of text written for

documentation (approximation based on 40 lines / page). Table 11 also shows that less than three classes per project month were completed on a project that is supposed to create lots of classes. The same can be seen on table 6, coding activity is merely bit over 10% of time spent.

2.6 Screenshots



Figure 7: Front page.



Figure 8: Event calendar.



Figure 9: Admin tools - add a new member.

2.7 Statistics

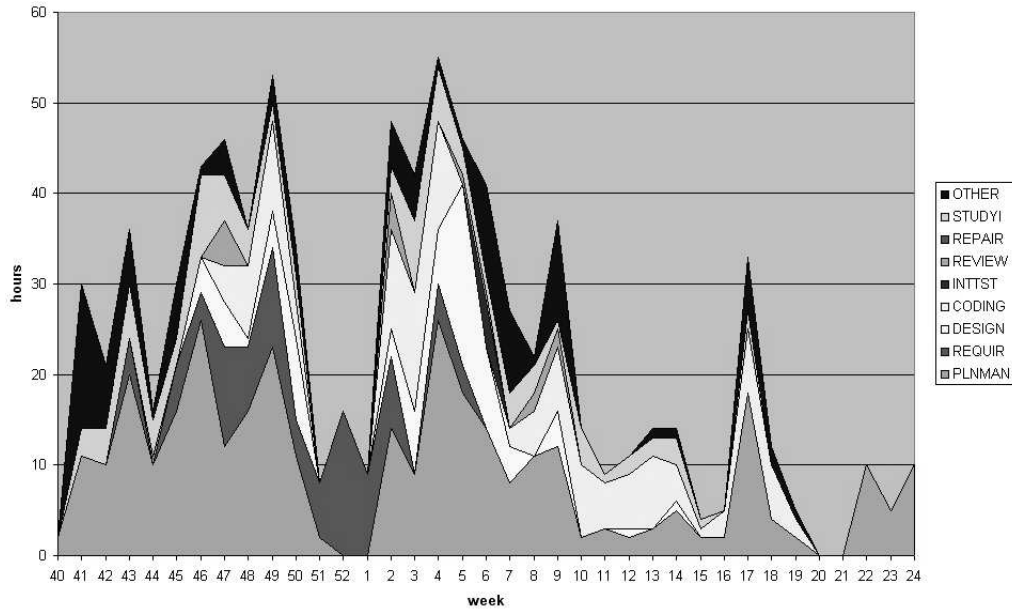


Figure 10: Weekly workload of the project by task.

Table 5: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
2+5+1	Waterfall	6.10.2007	27.5.2008	234	830	0.57

Table 6: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Design	Code	Integ. and test	Reviews	Repair	Study	Other	Total
Hours	339	91	78	126	5	16	0	88	101	844
%	40.2 %	10.8 %	9.2 %	14.9 %	0.6 %	1.9 %	0.0 %	10.4 %	12.0 %	100.0 %

Table 7: Requirements and high-level design outcome.

Pages	Requirements	Use-cases	UI screens	Database diagrams	Database tables
37	23	23	10	1	25

Table 8: Inspection findings.

	Project plan	Requirements specification	Implementation plan	Test plan
Pages and/or screens	43	37	14	12
Preparation time	0.5h	2h	1h	0.5h
Inspection time	1h	1h	1h	1h
Findings	5	10	5	9
Used time / findings	0.2	0.6	0.4	0.1

Table 9: Project's documents.

Document	Pages	versions
Preliminary analysis	5	1
Project plan	43	5
Requirements specification	37	7
User interface document	38	4
Test plan	3	12
Test report	tbd	1
Final report	19	4
Final story	6	1
Weekly reports	15	
Inspection reports	6	
Total	166	

Table 10: Project's codelines.

Language	PHP5
LOC	6069
SLOC	4206
Reused code	31571
Reused and modified code	
Classes	14
Functions	74
Code revisions	263

Table 11: Productivity metrics.

PM	LOC / PM	Classes / PM	RS & DES pages / PM	LOC pages / PM	Total pages / PM
5.46	1111	2.56	6.78	22.16	52.56

3 HAT 2007

3.1 Overview

HAT 2007 is follow up project continued the development of the software done by the projects HAT 2005 and HAT 2006. HAT software is a support system for heuristic evaluation. HAT 2007 project's goals were to add voting and statistic features and finalizes the product for the end user. HAT is a WWW-based application written in Java.

3.2 Organisation and management

The client of the project was the Usability Laboratory of the University of Tampere. The client's representatives were

- Saila Ovaska
- Harri Siirtola

The managers of the HAT 2007 project was

- Jaakko Leinonen
- Kai Stenvik

The members of the project group were

- Kirsikka Kaipainen
- Tommi Kallio
- Hanna Venesvirta (from the usability group)
- Minna Viitanen

Tommi Kallio and Kirsikka Kaipainen were the two primary persons responsible for coding the software. Minna Viitanen and Hanna Venesvirta were developing design and usability issues. One of the team members dropped out in the beginning of the project. Jaakko Leinonen and Kai Stenvik handled the Project Manager issues and the customer contacts. The group had weekly meetings during the project. Weekly report was sent to a customer and sponsor. Additionally group had an internet web-page where all the documents were posted and also different communications methods: IRC, email and phone. Group held multiple review meetings with the customer and a sponsor after every increment.

3.3 Methods and tools

The project group had to use the implementation techniques the previous group (HAT 2005) had chosen. PostgreSQL is the system's database and the data is manipulated with the help of Hibernate objects. In addition, Spring framework is used in building the system's architecture. The group thought that the implementation techniques were complicated and it took a lot of time to study their use. The group used Eclipse and Subversion (Subclipse) to help the coding and version management. The Subversion repository was held in a virtual server that was provided by the CS Department. The version management worked well, and it was helpful that all the project stakeholders could use the system in the internet.

3.4 Project phases and development model

In the beginning of the project the group spent a lot of time to getting to know the system and trying to figure out what the former team had done and what were the new requirements. Team had difficulties in the start to get the environment running. Two main obstacles in the course were the customer connection and the specification. It took very long time to get the approved specifications from the customer before we could start the actual work. Main implemented feature was the voting component and fixing the GUI layout to more customer wanted form.

3.5 Conclusions

The project was challenging but I think it gave the team good understanding of the "real" software project, customer issues, time pressure and different development tools and methods. It's more difficult to continue already started project with no background compared to one started from scratch. Advice to possible following project teams is, that try to lock the specs as early as possible so the actual development work can star ASAP, because it will take time get the knowledge of the special environment and the customer specification don't change all the time.

3.6 Sample screenshot

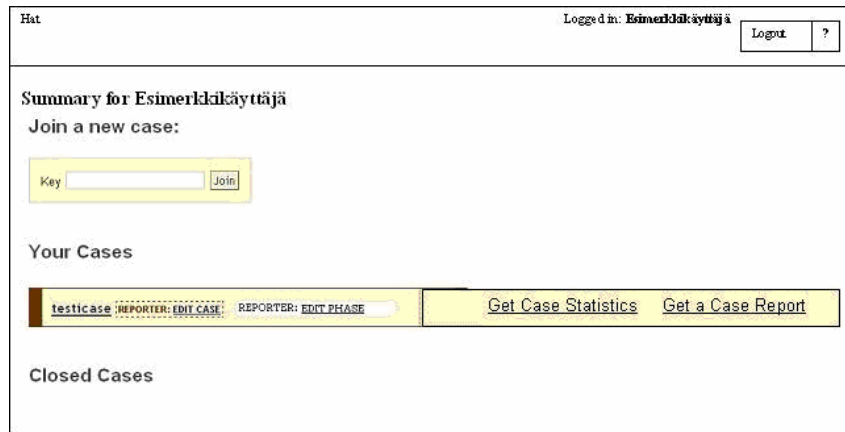


Figure 11: Main screen of the HAT.

3.7 Statistics

Table 12: Weekly working hours.

Week	Hours	Week	Hours	Week	Hours
41/2007	6	2/2008	34,5	15/2008	31
42/2007	42	3/2008	25,5	16/2008	14
43/2007	28	4/2008	18,5	17/2008	14
44/2007	12	5/2008	19	18/2008	19
45/2007	26,5	6/2008	20	19/2008	7,5
46/2007	26,5	7/2008	20,5	20/2008	4
47/2007	25	8/2008	32,5	21/2008	5
48/2007	39,5	9/2008	43,5	22/2008	5
49/2007	19	10/2008	53	23/2008	7
50/2007	16	11/2008	50	24/2008	18
51/2007	7	12/2008	47	TOTAL	833
52/2007	3,5	13/2008	39		
1/2008	20,5	14/2008	34		

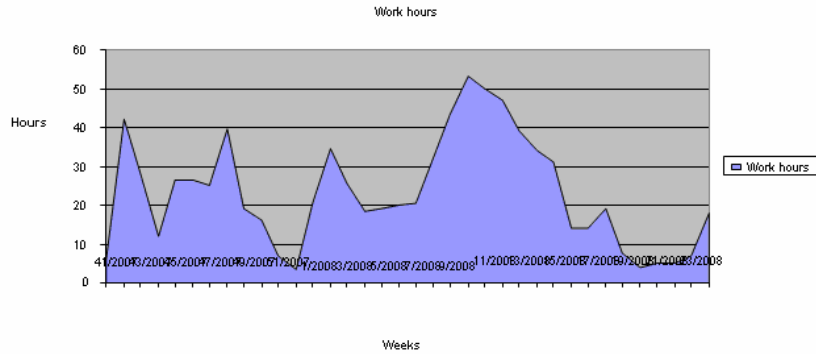


Figure 12: Work hours.

Table 13: General project information.

Phase	Start	End	Duration (week)
Project Start	41 (41)	49 (47)	9 (7)
Design	48 (47)	12 (5)	17 (11)
Iteration 1	10	18	9 (4)
Iteration II	12	18	7 (4)
Testing	18	21	4 (3)
Project end	21 (16)	24 (18)	4 (3)

Table 14: Project's documents.

Name	Version	Pages
Project plan	1.3	29
Specification plan (vaatimusmäärittely)	1.0	29
Test Plan	1.0	10
UI plan	1.4	28
Implementation plan	1.0	9
Testing report	1.0	8
Installation guide	1.0	3
Usability report	1.0	12
Final report	1.0	21
Final report (summary)	-	5
Total		153

4 WUH 2.0

4.1 Yleistä

Projektin tarkoituksena oli jatkokehittää Tampereen teknillisellä yliopistolla vuonna 2006 aloitettua koiranäyttelyiden hallintaohjelmaa. Koiranäyttelyiden hallintaohjelma WUH on sovellus, jonka avulla voidaan organisoida Suomessa järjestettäviä kansainvälisiä ja kansallisia koiranäyttelyitä sekä ryhmä- ja erikoisnäyttelyitä. Sovelluksen toimintoihin kuuluvat mm. näyttelyilmoittautumisten kirjaaminen, näyttelyiden kehäjärjestysten ja tuomarien organisoinnit, näyttelyluettelon laadinta sekä erilaisten raporttien laadinnat ja tulostukset. Sovelluksen tavoitteena on auttaa koiranäyttelyiden järjestämisessä ja niiden tulosten hallinnassa.



Figure 13: Ohjelman aloitusruutu.

4.2 Organisaation ja johtaminen

Projektiryhmä:

- Heikki Santasalo (heikki.santasalo@uta.fi)
- Antti Finni (antti.finni@uta.fi)

- Mervi Jokipii (mervi.jokipii@uta.fi)
- Tiina Taivalantti (tiina.taivalantti@uta.fi)
- Mikko Lindberg (mikko.lindberg@uta.fi) ja
- Arttu Tamminen (arttu.tamminen@uta.fi)

Heikki Santasalo ja Antti Finni vastasivat projektin johtamisesta ja hallinnollisista tehtävistä. Mervi Jokipii vastasi pääasiassa projektin web-sivuista sekä ohjelmoinnista. Tiina Taivalantti vastasi dokumentoinnista ja ohjelmoinnista. Mikko Lindberg vastasi testauksesta ja sen dokumentoinnista. Arttu Tamminen vastasi pääosin toteutuksesta. Edellä esitetyt roolit pyrittiin pitämään joustavina, joten periaatteessa kaikki ryhmän jäsenet vastasivat useammista osa-alueista.



Figure 14: Projektiryhmä.

Asiakas: Projektin asiakkaana on Satakunnan kennelpiiri ry:n neljä jäsenyhdistystä, joiden yhteyshenkilönä toimii Raine Kienokoski. Asiakas ja järjestelmän tilaaja on Raine Kienokoski (raine.kienokoski@saunalahti.fi).

4.3 Menetelmät ja työkalut

Alkuperäinen ohjelma eli WUH 1.0 on tehty käyttäen Java-ohjelmointikieltä, joten käytimme sitä myös versiossa 2.0. Javan version on oltava vähintään 1.5.0. Ohjelmiston käyttöliittymä toteutettiin Java Swing -teknologialla. Silta Java-luokkien ja tietokannan välille rakennettiin Hibernate-teknologialla. Hibernatesta käytetään 3.x -versiota. Tietokantaohjelmistona toimii HSQLDB

1.8.0, joka poikkeaa perinteisestä tietokantapalvelimesta siten, että se on tiedostopohjainen ja pyörii samassa prosessissa sovelluksen kanssa. Raporttien tekoon käytettiin iText-kirjastoa. Itse kehitys tapahtuu Eclipse 3.3 - IDEä käyttäen. Dokumentit kirjoitimme pääasiallisesti MS Word-tekstinkäsittelyohjelmalla.

4.4 Projektin vaiheet ja kehitysmalli

Projektin kehitysmalli oli inkrementaalisen kehitysmallin ja vesiputousmallin yhdistelmä. Projekti aikataulutettiin alunperin vesiputousmallin mukaan, mutta kehitysvaiheen ollessa huomattavan työläs, otettiin siihen avuksi inkrementit. Inkrementtien avulla projektin eteenmistä voitiin valvoa paremmin.

Projektiryhmän kesken pidettiin viikkopalaveri joka viikko, poislukien muutamat juhlapyhien ajat. Muutamina viikkoina kokonnuttiin useamman kerran tarpeen niin vaatiessa. Jokaisen viikkopalaverin osalta kirjoitettiin viikkoraportti ja se lähetettiin kaikille sidosryhmille. Katselmointeja järjestettiin sekä kurssin vetäjän, että asiakkaan kanssa. Ryhmä tarkasti dokumentteja myös sisäisesti niiden muuttuessa inkrementaalisen kehitysmallin myötä usein. Dokumenttien ylläpito oli käytännössä edellisen projektityöryhmän dokumenttien hyödyntämistä ja päivittämistä. Projektin aikana pidettiin viikkopalaverien lisäksi workshoppeja, mutta kehitystyökalujen tullessa tutummaksi workshoppien tarpeellisuutta ei koettu kovien tärkeäksi, vaan projektia vietiin eteenpäin omalla ajallaan. Tämä tuntui toimivan lopulta hyvin koska aikataulujen sovittaminen saattoi olla haastavaa koska monet ryhmäläisistä olivat työssäkäyviä. Projektin oma IRC-kanava koettiin hyväksi kommunikaatiokanavaksi.

4.5 Yhteenveto

Kurssia pidettiin osittain opettavaisena, mutta todettiin myös että projektityökurssia ei voi verrata työelämään. Interessit, kannustaminen, motivoituneisuus ja ajankäyttö ovat kuitenkin aivan eri perusteisia. Koska kyse oli jatkokehitysprojektista, oli toteutuksen arkkitehtuuriin sisäänpääseminen haasteellista, mutta toisaalta vaatimusmäärittely oli periaatteessa valmiina. Näinkin paljon toiminnallisuutta sisältävän ohjelmiston jatkokehitys on jokapäiväisessä haasteellista. Pienistä vastoinkäymisistä huolimatta projektia voidaan pitää onnistuneena. Lopputuote vastaa hyväksyttyä vaatimusmäärittelyä, vaikka kaikkiin asiakkaan toiveisiin ei voitu enää vaikuttaa.

4.6 Statistics

Table 15: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
2+4+1	Incremental	15.10.2007	14.5.2008	212	1200	0.943

Table 16: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Des-ign	Code	Integ. and test	Rev-iews	Re-pair	Study	Other	Total
Hours	368	0	0	559	45	47	0	181	0	1200
Usability	0	0	0	0	0	0	0	0	0	0
Total	368	0	0	559	45	47	0	181	0	1200

Table 17: Requirements and high-level design outcome.

Pages	Requirements	Use-cases	UI screens	Database diagrams	Database tables
	84	21	35	1	24

Table 18: Design outcome.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
	5	2	0	0	5

Table 19: Inspection findings.

	Project plan	Inspection 1	Inspection 2	Inspection 3
Pages and/or screens	121	0	0	0
Preparation time	17	10	10	10
Inspection time	6	6	6	6
Findings	25	15	15	15
Used time / findings	0	0	0	0

Table 20: Project's documents.

Document	Pages	versions
Preliminary analysis	6	
Project plan	20	
Project's usability plan	10	
Requirements specification	121	
Design plan	170	
User interface document	0	
Test plan	100	
User's guide	170	
Test report	5	
Usability test report	0	
UI Heuristic evaluation report	0	
Final report	17	
Final story	5	
Weekly reports	28	
Inspection reports		
Total		

Table 21: Project's codelines.

Language	JAVA
LOC	5000
SLOC	4000
Reused code	0
Reused and modified code	0
Classes	103
Functions	0
Code revisions	30

Table 22: Productivity metrics.

PM	LOC / PM	Classes / PM	RS & DES pages / PM	LOC pages / PM	Total pages / PM
	641	13	37	12,8	721

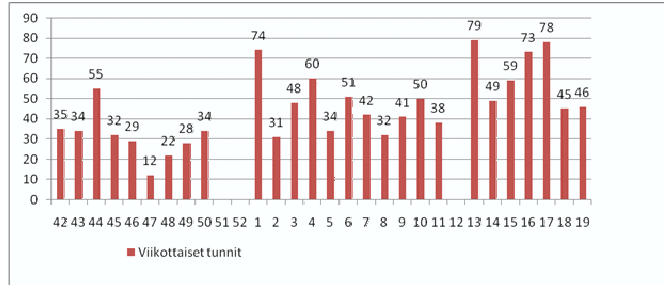


Figure 15: Viikottaiset työtunnit.

Kaavio 4. Henkilökohtaiset tunnit viikottain

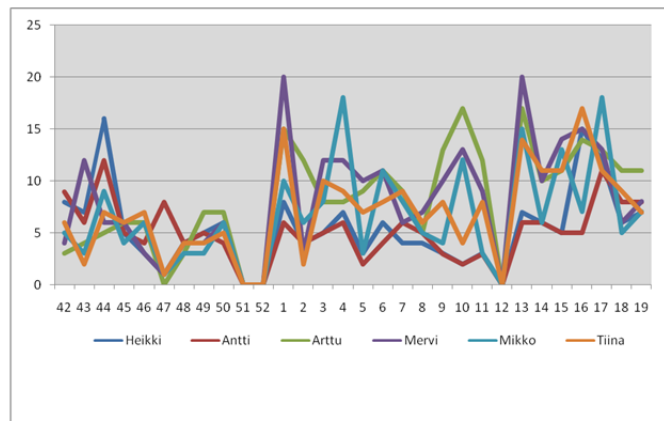


Figure 16: Henkilökohtaiset tunnit viikottain.

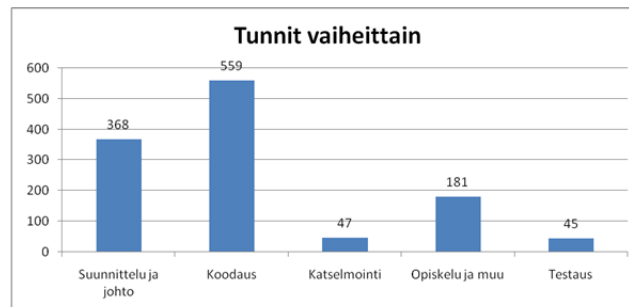


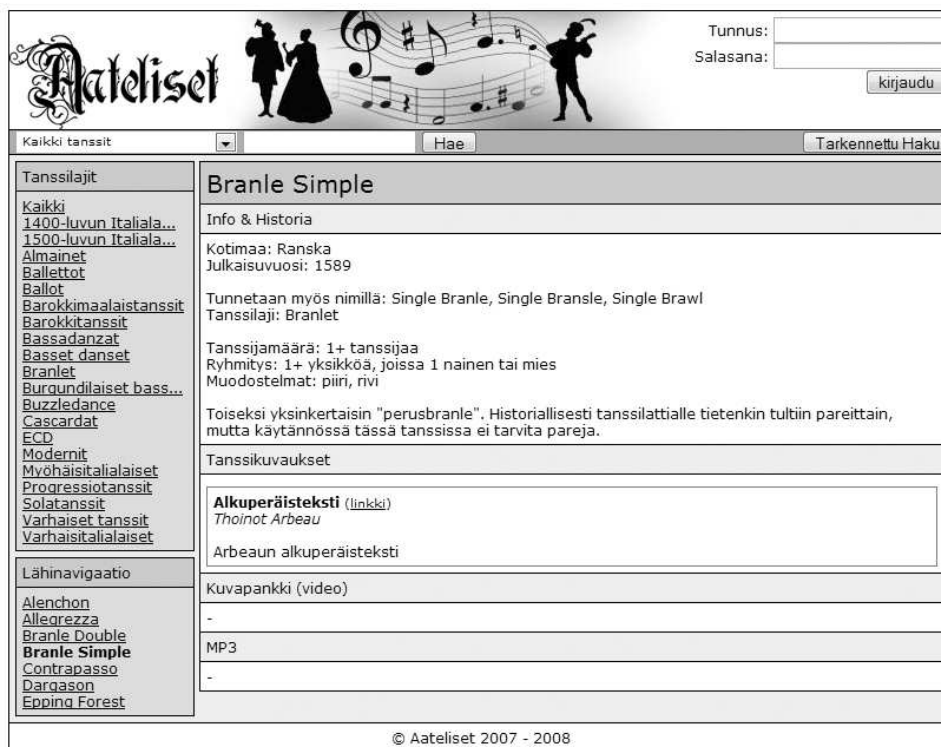
Figure 17: Tunnit projektin vaiheittain.

5 Aateliset

5.1 Johdanto

Projektissa oli tarkoituksena tuottaa tietokanta renessanssitansseja varten sekä www-pohjainen käyttöliittymä tälle tietokannalle. Valmiiseen tietokantaan on mahdollista sekä dokumentoida, että hakea ja selata erilaisia tansseja. Tietokannan informaatio sisältää tanssin mahdolliset variaatiot, askelkuvioita ja alkuperäistekstejä. Kaikki näistä ominaisuuksista eivät ole relevantteja jokaisen tanssin osalta, mutta jokaisesta tanssista on vähintään perustiedot eli nimi ja askelkuviot.

Asiakkaana on Hämeen keskiaikaseura, jota edustavat tanssimestari Atro Kajaste ja jäsen Ilari Kajaste.



The screenshot shows the Aateliset website interface. At the top, there is a header with the logo 'Aateliset' and silhouettes of dancers. Below the header, there is a search bar with a dropdown menu set to 'Kaikki tanssit', a search button 'Hae', and a 'Tarkennettu Haku' button. On the right side of the header, there are input fields for 'Tunnus:' and 'Salasana:', and a 'kirjaudu' button. The main content area is divided into two columns. The left column contains a 'Tanssilajit' menu with a list of dance categories such as 'Kaikki', '1400-luvun Italia...', 'Almainet', 'Ballettot', 'Ballot', 'Barokkimaalaistanssit', 'Barokkitanssit', 'Bassadanzat', 'Basset danset', 'Branlet', 'Burqundilaiset bass...', 'Buzzedance', 'Cascardat', 'ECD', 'Modernit', 'Myöhäisitalialaiset', 'Progressiotanssit', 'Solatanssit', 'Varhaiset tanssit', and 'Varhaisitalialaiset'. Below this is a 'Lähinavigaatio' section with links to 'Alenchon', 'Allegrezza', 'Branle Double', 'Branle Simple', 'Contrapasso', 'Dargason', and 'Epping Forest'. The right column displays the search result for 'Branle Simple'. It includes an 'Info & Historia' section with details: 'Kotimaa: Ranska', 'Julkaisuvuosi: 1589', 'Tunnetaan myös nimillä: Single Branle, Single Bransle, Single Brawl', and 'Tanssilaji: Branlet'. It also lists 'Tanssijamäärä: 1+ tanssijaa', 'Ryhmyitys: 1+ yksikköä, joissa 1 nainen tai mies', and 'Muodostelmat: piiri, rivi'. A descriptive paragraph follows: 'Toiseksi yksinkertaisin "perusbranle". Historiallisesti tanssilattialle tietenkin tultiin pareittain, mutta käytännössä tässä tanssissa ei tarvita pareja.' Below this is a 'Tanssikuvaukset' section with a link to 'Alkuperäisteksti (linkki)' and the text 'Thoinot Arbeau' and 'Arbeaun alkuperäisteksti'. There are also sections for 'Kuvapankki (video)' and 'MP3', both showing a dash '-'. At the bottom of the page, there is a copyright notice: '© Aateliset 2007 - 2008'.

Figure 18: Käyttöliittymä

5.2 Organisaatio

Projektipäällikköinä toimivat Virpi Tuohisto ja Päivi Urpelainen.

Ohjelmoinnista ja osasta suunnittelua vastasivat Harri Heinisuo ja Matti Virtanen. Tuotteen käytettävyydestä, suunnittelusta ja toiminnallisuustestauksesta vastasivat Jussi Hautaniemi, Tony Keisala ja Ilari Valta. Graafisesta ulkoasusta vastasi Jussi Hautaniemi. Käytettävyydestien järjestämiseen osallistuivat käytettävyysryhmän jäsenen Jenni Päckilän lisäksi Jussi, Tony ja Ilari.



Figure 19: Projektipäälliköt Päivi ja Virpi



Figure 20: Projektiryhmäläiset Harri, Ilari, Jussi ja Tony

5.3 Menetelmät ja käytetyt ohjelmistot

Projektin kehitysvaiheessa käytettiin yleisesti käytössä olevia tekstieditoreja ja uusimpia selaimia.

Kehitysmallina projektissa käytimme Agile-menetelmiin kuuluvaa Scrumia. Scrum osoittautui oikeaksi valinnaksi projektimme luonteen takia; asiakas esitti toiminnallisuusvaatimuksia vaiheittain projektin edetessä, toteutuksen edetessä tuotteemme myös loi uusia ideoita ja toteutusvaatimuk-

sia. Useimmissa suunnittelupalavereissa käytimme aika-arvioinnissa Agile-menetelmiin hyvin sopivaa Planning pokeria.

5.4 Projektin eteneminen

- Ensimmäinen ryhmätapaaminen 17.10.2007:
Projektin lähti liikkeelle ryhmän jäsenten esittäytymisellä ja tutustumisella. Alustavasti kartoitettiin myös jäsenten taidot ja tehtiin työnjakoa. Vireille laitettiin ryhmän ja projektin nimen ja logon suunnittelu. Keskustelimme projektin vaiheista, tavoitteista ja päämääristä sekä mahdollisista riskeistä. Valitsimme kehitysmalliksemme Agile-menetelmiin kuuluvan Scrummin ja jaoimme koko projektin 4-5 sprinttiin, joista ensimmäisen suoritamme ennen vuodenvaihdetta.
- Ensimmäinen asiakastapaaminen 31.10.2007:
Kävimme läpi asiakkaiden toiveita tanssitetokannan, lopullisen tuotteen ja projektin suhteen. Projektin päämäärät ja tavoitteet selkenevät.
- Ryhmätapaaminen 31.10.2007:
Aateliset-logo ja nimi hyväksytään projektille, www-sivun ja tuntikirjanpidon toteutuksesta sovitaan. Pyrimme pitämään sisäisen katselmoinnin noin viikkoa ennen virallisia katselmointeja tai asiakasdemoja.
- Esitutkimuksen katselmointi 7.11.2007:
Katselimme esitutkimuksen. Keskustelimme renessanssitansseista ja siitä, mihin asiakas tietokantaa oikeasti tarvitsee. Selvittelyä Scrum-prosessin käytöstä.
- Ryhmätapaaminen 14.11.2007:
Kävimme läpi projektisuunnitelman. Lisäsimme ensimmäisille tehtäville aika-arviot ja kokosimme ensimmäisen tehtäväluettelon seuraavalla viikolla alkavaa sprinttiä varten.
- Ryhmätapaaminen 21.11.2007:
Projektisuunnitelman toinen vaihe käytiin läpi. Selvitimme muutokset, joita on tullut toiminnallisuusluetteloön asiakaskeskustelujen myötä. Keskustelimme tulevista tehtävistä. Aloitimme ensimmäisen sprintin.
- Projektisuunnitelman katselmointi 29.11.2007

- Ensimmäisen asiakasdemo 12.12.2007:
Demosimme aikaansaannoksiamme ja keskustelimme projektista. Ensimmäinen sprintti loppui.
- Ryhmätapaaminen 9.1.2008:
Toinen sprintti alkaa. Selvitimme aikataulun ja tehtävät uuteen sprinttiin. Selvitimme, mitä projektissamme pitää kehittää ja parantaa.
- Toinen asiakasdemo 29.1.2008:
Demosimme aikaansaannoksiamme ja keskustelimme projektista. Toinen sprintti loppui.
- Ryhmätapaaminen ja demo 6.2.2008:
Pidimme pienimuotoisen demon Timo Poraselle ja selvitimme projektimme tilaa. Kävimme läpi edellistä sprinttiä ja suunnittelimme tulevaa. Aloitimme uuden sprintin.
- Kolmas asiakasdemo 4.3.2008:
Esittelimme uutta toiminnallisuutta. Keskustelimme asiakkaan toiveista ja kehitysohjon liittyvistä kysymyksistä. Sovimme käytettävyydestä tarkemmin.
- Käytettävyydestejä 12.3.2008-13.3.2008
- Hakupalaveri 17.3.2008:
Kävimme läpi hakuun liittyviä asioita asiakkaiden kanssa.
- Ryhmätapaaminen 19.3.2008:
Kävimme läpi viimeaikaisia tapahtumia, pidimme sprintin palautekeskustelun ja suunnittelimme tulevaa. Aloitimme neljännen sprintin, jossa tärkeimpiä tehtäviä ovat käytettävyys- ja toiminnallisuustestaus, dokumentaatio ja toteutuksen viimeistely.
- Käytettävyydestejä 19.3.2008
- Käytettävyysraportti 4.4.2008:
Käytettävyysraportin valmistuminen.
- Päätöspäivä tietokannan kehitysohjon 15.4.2008
- Ryhmätapaaminen 16.4.2008:
Aloitetaan loppuraportin tekeminen ja projektin viimeistely.

- Loppukertomuksen palautus, projektin esittely 7.5.2008
- Viimeinen asiakastapaaminen sekä projektin lopetus 14.5.2008:
Palautetaan projekti-cd kaikkine dokumentteineen ja päätetään projekti.

Muutoksia projektin kulkuun tuli lähinnä aikataulullisissa suunnitelmissa. Projekti oli alun perin kaavailtu jaettavaksi viisi sprinttiä käsittäväksi kokonaisuudeksi, mutta kolmannen sprintin loppupuolella näimme järkeväksi tiputtaa yhden sprint-kierroksen pois ja venyttää viimeisiä kierroksia alkuperäisistä. Samoin alun perin kaavailut päivämäärät asiakasdemojen kohdalla muutuivat asiakkaan toivomuksesta. Työskentely painottui usein sprintin loppupuolelle, vaikka tarkoitus oli jakaa työmääriä tasaisesti koko sprintin ajalle.

5.5 Johtopäätökset

Mielestämme projekti oli hyvin onnistunut kokonaisuutena. Ryhmän yhteistyö toimi, kaikki olivat aktiivisesti mukana ja motivaatiota riitti koko pitkään projektiin. Tuntui, että jokaiselle löytyi mielekäs vastuualue ja rooli projektissa. Projektin eteneminen oli jatkuvaa. Pääosin pysyimme tavoitteissamme, vaikka lopussa hieman aikataulu pettikin. Työskentely sprintin sisällä painottui usein sen loppupuoliskolle, mutta tehtävät tulivat kuitenkin tehtyä. Jatkossa voisi yrittää aikaistaa tehtävän aloittamista, jolloin työmäärä jakaantuisi tasaisemmin koko sprintille. Toinen vaihtoehto olisi sprinttien lyhentäminen esimerkiksi kahteen viikkoon. Valittu kehitysmalli, välineet ja menetelmät sopivat projektiimme.

5.6 Tilastot

Koska projektimme seurasi Scrum-menetelmää, myös statistiikka eroaa vesiputousmallista; Agile-menetelmissä esimerkiksi toimivaa lopputuotetta arvostetaan enemmän kuin muodollista dokumentaatiota.

Table 23: Yleistietoa projektista.

Ryhmä- koko	Keh. malli	Aloitus pvm	Lopetus pvm	Päivät	Tunnit	Tunnit / (Päivät*Ryhmä)
2+5+1	Scrum	17.10.2007	14.5.2007	211	1221,5	0.72

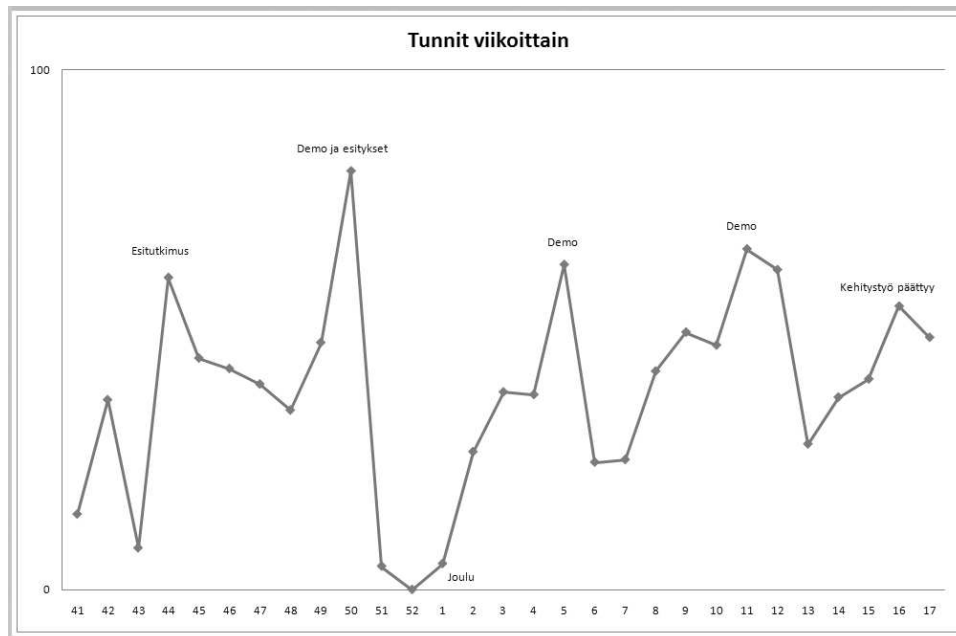


Figure 21: Työtunnit viikoittain.

Table 24: Työtunnit aihealueittain.

Kategoria	Proj. suunn. ja joht.	Vaativ. määr.	Suunnitelu	Tot.	Integ. ja testaus	Korj.	Katselointi	Opisk.	Muut	Yht.
Tunnit	322,5	7,5	287	213	54,5	37,5	9	116,5	108,5	1156
%	27,9	0,7	24,8	18,4	4,7	3,2	0,8	10,1	9,4	100
Käytetty										65,5
Yht.										1221,5

Projektissamme suunnitelmat elivät jatkuvasti perustuen muun muassa asiakaspalautteeseen, näin ollen varsinainen suunnitteludokumentaatio jäi melko kevyeksi.

Vaatimusten sydän projektissamme oli Exceliin sijoitettu toiminnallisuusluettelo. Se sisälsi 40 korkean tason vaatimusta, joista 19 toteutettiin.

Table 25: Vaatimukset ja korkean tason suunnitelmat.

Sivuja	Vaatimuksia	Käyttötapauksia	Käyttöliittymänäkymiä	Tietokanta-kaavioita	Tietokantatauluja
1	41	5	7	14	

Kehitysmenetelmästäemme johtuen suurin osa katselmoinnista oli asiakkaalle järjestettyjä demoja, joista löytyy lisätietoa kappaleesta Projektin eteneminen.

Table 26: Katselmointilöydökset.

	Projektisuunnitelma
Sivuja	18
Valmistautumisaika	720
Katselmointiaika	480
Löydöksiä	33
Käytetty aika/löydökset	33,4

Table 27: Dokumentaatio.

Dokumentti	Sivuja	Versioita
Esitutkimus	10	1
Projektisuunnitelma	18	5
Käytettävyyssuunnitelma	6	1
Käytettävyytestiraportti	18	11
Loppuraportti	18	1
Loppukertomus	7	1
Viikkoraportit	29	
Katselmointiraportit	8	
Yhteensä	114	

Table 28: Koodi.

Kieli	PHP
LOC	3978
SLOC	3659
Tiedostoja	10
Funktioita	51

Table 29: Tuottavuus.

PM	LOC / PM	Tiedostot / PM	LOC sivut / PM	Kaikki sivut / PM
	495,01	1,24	9,90	24,09

6 TILATAR

6.1 Yleistä

TILATAR-projektin tehtävänä oli toteuttaa Tampereen Työväenyhdistys ry:lle Tampereen työväentalon kokous-, ryhmätyö- sekä juhlatilojen ja näiden tilavausten hallinnointiin sopiva monipuolinen varauskalenterijärjestelmä. Lopputuote on helppokäyttöinen web-pohjainen sovellus, jota käyttävät niin Tampereen työväentalon tilojen hallinnoijat - eli lähinnä Tampereen Työväenyhdistyksen työntekijät - kuin tilojen käyttäjät, eli työväentalon tiloissa toimivat erilaiset yhdistykset ja yhteisöt.

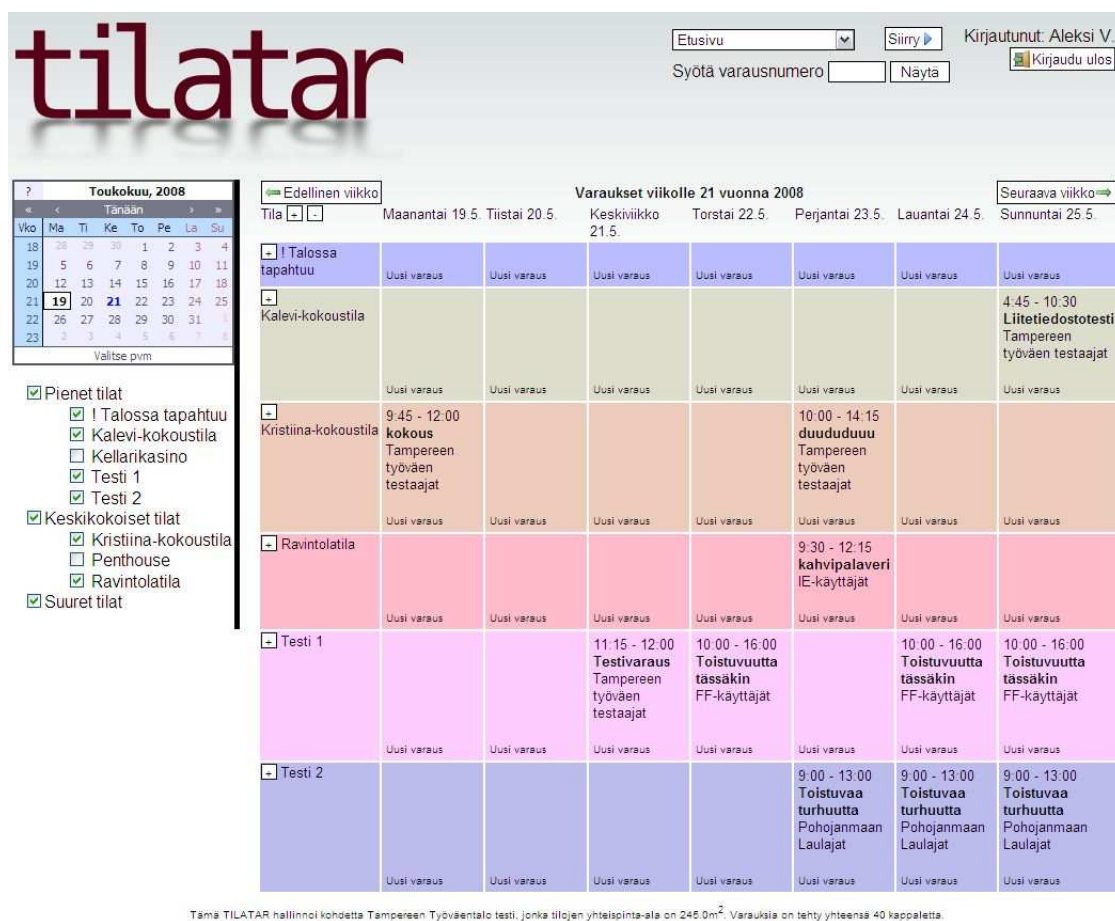


Figure 22: Näytönkaappaus TILATAR-järjestelmän päänäkymästä, viikkokalenterista.

tilatar

Etusivu | Kirjautunut: Aleksi V. | Kirjautuu ulos

Syötä varausnumero | Näytä

Perustiedot | Lisätiedot | Vahvistus

Varauksen tekijä: Aleksi V.

Varauksen tehnyt organisaatio: Tampereen työväen testajaat

Vastuuhenkilö: Ville Vastuullinen

Varauksen otsikko (pakollinen): Kevätkokous

Tila: Kristina-kokoustila

Tilan varaustilanne: 10:00 - 14:15 duududuu

Varausaika: Alkamisaika: 23.05.2008 16:00 | Päätymisaika: 23.05.2008 18:00

Toistopäivämäärät: 2008-05-23 16:00:00

Erityistoiveet: Tarvitsemme muutaman ylimääräisen tuolin kokousvieraita varten, toivottavasti onnistuu.

Liitetiedostot: TILATAR_Asiakirjamalli_v0-1.dot - 163 kilotavua | TILATAR_Vaatusmääritys_v1-0.pdf - 502 kilotavua

Selaa | Lähetä

Jatka

Figure 23: Näytönkaappaus TILATAR-järjestelmästä, uuden varauksen tekeminen.

6.2 Projektioorganisaatio

Projektin asiakkaana oli Tampereen Työväenyhdistys ry., jonka edustajina toimivat Matti Länsiö, toiminnanjohtaja, sekä Jussi Koskinen, kiinteistöhoitaja.

Projektiryhmään kuului viisi jäsentä (suluissa päävastualueet): Aleksi Vuorenmaa (projektipäällikkö), Heidi Kari (testaus, ohjelmointi ja dokumentointi), Markus Laurila (testaus, ohjelmointi ja käyttöliittymän toteutus), Iikka Mattila (ohjelmointi ja arkkitehtuurisuunnittelu) sekä Reetu Mönkkönen (ohjelmointi, tietokanta ja käyttöliittymän toteutus).

Lisäksi ryhmään kuuluivat myös käytettävyysryhmän jäsenet Matti Huokko ja Santtu Mansikkamaa, joiden vastualueena oli käyttöliittymän suunnit-

telu ja käytettävyysestaus.



Figure 24: Projektiryhmän jäsenet Reetu (vas.), Markus, Iikku, Heidi, Aleksi, Matti. Kuvasta puuttuu Santtu.

Alun perin projektiryhmän muodostumisen jälkeen ryhmän kokoa muutettiin yhteensä kaksi kertaa kahden projektiryhmän jäsenen jätettyä kurssi kesken.

Projektiryhmä kokoontui viikoittain laitoksen projektityöhuoneessa, muutamia poikkeuksia, kuten joului- ja pääsiäislomaa, lukuunottamatta. Yhteydenpitokanavana käytettiin lisäksi sekä sähköpostilistaa että projektiryhmän IRC-kanavaa #tilatar. Viikkoraportteja lähetettiin asiakkaan edustajille sekä kurssin valvojille yhteensä 10 kpl.

6.3 Metodit ja työkalut

Järjestelmä on toteutettu Java 1.5-versiolla ja se toimii Apache/Tomcat-pohjaisella palvelimella. Järjestelmän toteutuksessa käytettiin erittäin oleellisenä osana Spring Framework -ohjelmistokehystä ja yhteys MySQL-tietokannasta sovellukseen on toteutettu Java Database Connectivity API:lla. Järjestelmään tallennettujen tietojen vientityökalun toteutuksessa hyödynnettiin jXLS-Javakirjastoa.

Projektissa käytettiin apuna seuraavia työkaluja:

- NetBeans IDE 5.5.1
- Subversion ja NetBeans SVN-plugin
- phpMyAdmin 2.7.0
- MediaWiki
- Google Docs & Spreadsheets
- Google Groups

- OpenOffice.org
- Microsoft Word

6.4 Projektin vaiheet

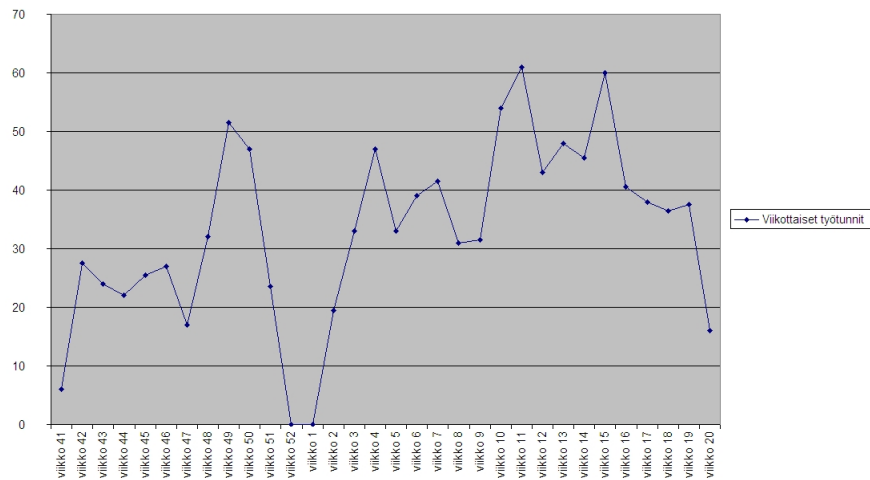


Figure 25: Weekly workload of the project.

Projektin kehitysmalliksi valittiin inkrementaalinen kehitysmalli. Projekti suunniteltiin jaettavaksi kolmeen inkrementtiin. Vaatimusmäärittelyt suunniteltiin tehtäväksi kaikille kolmelle inkrementille kerralla. Jälkeenpäin arvioiden voidaan todeta, että toteutunut kehitysmalli muistutti ehkä kuitenkin enemmän vesiputousmallia, sillä inkrementteillä ei ollut selviä itsenäisiä kokonaisuuksia, vaan vaatimusten toteutuksessa edettiin vain parhaassa ja prioriteettijärjestyksessä eteenpäin.

Projektin tärkeimmät vaiheet:

- 15.10.2007: Projektiryhmän ensimmäinen tapaaminen
- 23.10.2007: Ensimmäinen asiakastapaaminen
- 04.12.2007: Projektisuunnitelman katselmointi
- 19.12.2007: Vaatimusmäärittelyn katselmointi
- 14.03.2008: Inkrementtien 1 ja 2 katselmointi
- 18.05.2008: Ohjelmoinnin päättäminen
- 22.05.2008: Projekti-cd:n toimittaminen

6.5 Johtopäätökset

Monelle projektin jäsenistä projekti oli ensimmäinen oikea ohjelmistoalan työprojekti. Projekti opetti uusia asioita ja antoi ryhmän jäsenille käsityksen siitä, minkälaisia työelämän ohjelmistoprojektit todennäköisesti voivat olla. Ryhmän jäsenet saivat myös kokemusta projektissa työskentelemisestä ja saivat mahdollisuuden soveltaa aiemmin opittuja taitoja käytännössä.

Jos projekti aloitettaisiin nyt alusta, suurin etu olisi kehitysympäristön ja työkalujen tuntemus - monet ratkaisuihin veisivät huomattavasti vähemmän aikaa ja olisivat yhdenmukaisia muiden vastaavien ratkaisujen kanssa. Nykyisellään samoja koodiongelmia on ratkottu usealla eri tavalla.

Projekti oli hyvin organisoitu: säännölliset ryhmäpalaverit pitivät projektin hienosti kasassa ja niissä asetut osatavoitteet tulivat ajallaan toteutetuiksi. Myös ryhmän sisällä syntynyt selkeä työnjako osaamisen mukaan - pienen alkuvaiheen hakemisen jälkeen - auttoi saamaan asiat ajallaan valmiiksi. Asiakas oli kiinnostunut projektista ja osallistui kehitykseen.

Projekti onnistui, koska sovellus on käyttökelpoinen sekä asiakkaan vaatimusten mukainen ja näin ollen asiakas oli myös tyytyväinen lopputulokseen.

6.6 Tilastoja

Table 30: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
1+4+2	INC/WF	15.10.2007	22.5.2008	220	1014,50	0,66

Table 31: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Design	Code	Integ. and test	Reviews	Repair	Study	Other	Total
Hours	366	26,5	14,5	342,5	72	36	0	47,5	45	950
%	38,53%	2,79%	1,53%	36,05%	7,58%	3,79%	0%	5,00%	4,74%	100%
Usability										64,5
Total										1014,50

Table 32: Requirements and high-level design outcome.

Pages	Requirements	Use-cases	UI screens	Database diagrams	Database tables
65	21	7	21	1	18

Table 33: Design outcome.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
59		5			1

Table 34: Inspection findings.

	Project plan	Requirements specification	Increments 1 & 2
Pages and/or screens	35	32	98
Preparation time	10h	10h	10h
Inspection time	20h	20h	20h
Findings	18	24	19
Used time / findings	1,67	1,25	1,58

Table 35: Productivity metrics.

PM	LOC / PM	Classes / PM	RS & DES pages / PM	LOC pages / PM	Total pages / PM
6,67	2293,25	19,49	18,59	45,87	88,29

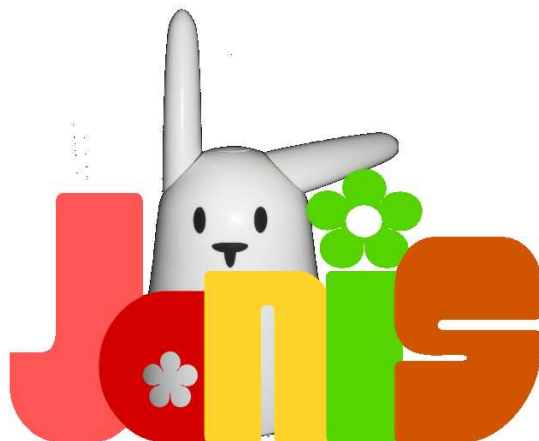
Table 36: Project's documents.

Document	Pages	versions
Preliminary analysis	8	1
Project plan	35	4
Project's usability plan	11	2
Requirements specification	32	3
Design plan	26	8
User interface document	33	10
Test plan	39	7
User's guide	32	5
Test report	7	3
Usability test report	11	4
UI Heuristic evaluation report	3	1
Final report	28	2
Final story	4	2
Weekly reports	10	
Inspection reports	7	3
Total	283	55

Table 37: Project's codelines.

Type	Files	LOC	SLOC
Java	39	7354	4492
JavaServer Pages	46	3985	3859
JavaScript	15	2557	2183
CSS	2	1400	1147
Picture files	12		
XML	8		
Properties files	4		
XLS templates	3		
MF configuration files	1		
Total	130	15296	11681

7 Janis



7.1 Yleiskuvaus projektista

Janis-projektin (**J**ava-**N**abaztag **I**nteractive **S**ystem) tarkoituksena oli tutkia julkista ja puolijulkista vuorovaikutusta kehittämällä prototyyppejä, jotka käyttävät kommunikoinnissa fyysistä, multimodaalista Nabaztag/tag -agenttia. Projektin aikana laadittiin agenttia käyttäviä vuorovaikutusskenaarioita, joiden pohjalta toteutettiin yksi prototyyppi, konferenssikaniijärjestelmä.

Konferenssikaniijärjestelmä on tarkoitettu konferenssien ja muiden yleisten tapahtumien käyttöön. Järjestelmän avulla konferenssin järjestäjät voivat kerätä palautetta erilaisista konferenssiin liittyvistä tapahtumista. Nabaztag/tag-kaneja sijoitetaan eri puolille tapahtuma-aluetta ja ihmiset voivat itsenäisesti käydä antamassa palautetta tapahtumista. Palautteenanto tapahtuu vastaamalla kanin verbaalisesti esittämiin kysymyksiin RFID-tagein; kanelle annetaan ”nuuhkittavaksi” RFID-tagilla varustettu objekti, joka kuvaa henkilön mielipidettä esitettyyn kysymykseen. Objektit ovat konferenssin järjestäjien vapaasti valittavissa. Tämän lisäksi konferenssin järjestäjillä on käytössään hallintaohjelma (Kuva 26), jolla pystytään muokkaamaan tapahtumia ja niihin liittyviä kysymyksiä.

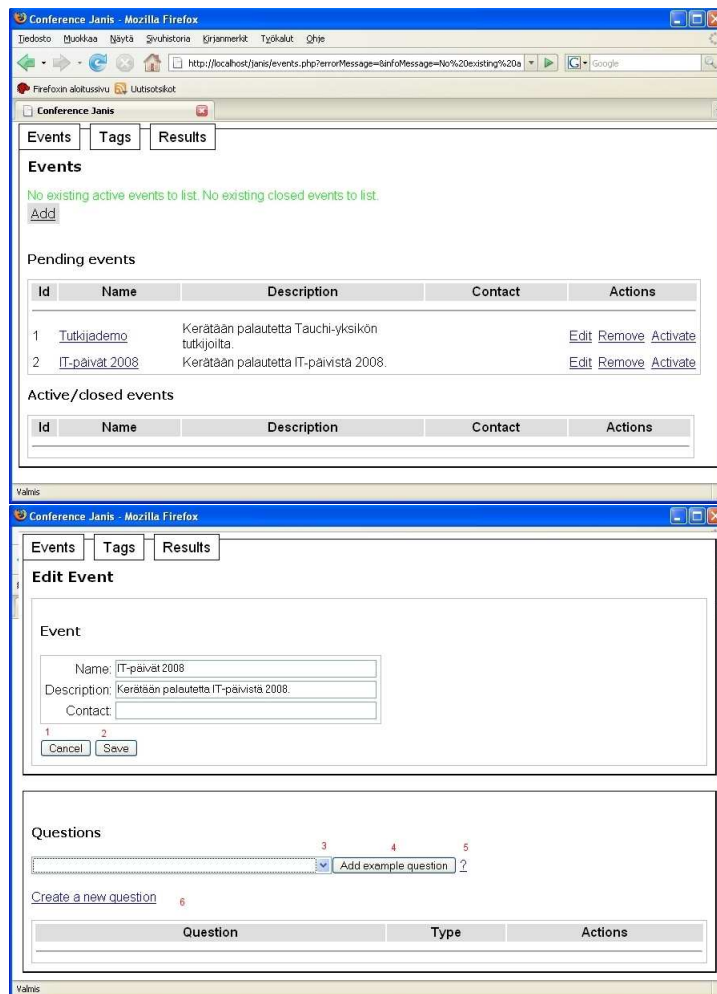


Figure 26: Konferenssikanin hallintaohjelma.

7.2 Organisaatio ja projektin hallinta

Projektissa työskenteli kaiken kaikkiaan yhdeksän henkilöä. Projektipäälliköinä toimivat Harri Pirttinen ja Pellervo Valkama. Projektiryhmäläisiä olivat Linda Andersson, Ville Antila, Susanna Asunmaa, Henri Kumanto sekä Jouni Vaaramo. Lisäksi ryhmään kuului kaksi käytettävyyssryhmän jäsentä, Johanna Huhtala ja Hanna Venesvirta.

Projektiryhmä pyrki kokoontumaan viikottain viikkopalaveriin, jossa käytiin läpi projektin eri osa-alueiden edistymisen edellisen viikon aikana sekä käsiteltiin projektin tulevia asioita ja laadittiin toimintasuunnitelma seuraavalle



Figure 27: Janis-ryhmä ja Nabaztag/tag-kani.

viikolle. Projektiryhmäläiset pitivät myös mm. erillisiä suunnittelu- ja implementaatiotapaamisia. Projektipäälliköt kokosivat viikon lopussa viikon aikana tapahtuneet asiat yhteen ja laativat niiden pohjalta viikkoraportin, joka toimitettiin projektin ohjausryhmälle sekä projektin jäsenille. Lisäksi projektipäälliköt seurasivat projektin edistymistä laatimalla iteraation raportin aina iteraation päätyttyä.

Projektin edetessä pidettiin projektin tuotoksista katselmoiteja. Katselmoiteja pidettiin sekä projektiryhmän sisällä, jolloin ryhmän kesken käytiin toteutettuja dokumentteja läpi, että ohjausryhmän kanssa, jolloin projektissa toteutettuja dokumentteja tarkasteltiin asiakkaan sekä kurssin vastuhenkilöiden kanssa. Katselmoineista saadun palautteen perusteella tehtiin korjauksia dokumentteihin sekä projektin suuntaan.

7.3 Menetelmät ja työkalut

Järjestelmä toteutettiin PHP- ja Java-kielillä. Lisäksi kehitystyössä ja sen apuna käytettiin seuraavia työkaluja:

- Fujaba

- Eclipse
- MySQL
- Subversion
- StarUML
- Subclipse
- TortoiseSVN

Projektissa käytettiin ketterää ohjelmistonkehitysmallia. Järjestelmän toteuttamisessa pyrittiin mahdollisuuksien mukaan käyttämään olioperustaisen suunnittelun ja toteutuksen periaatteita, jotta järjestelmän ylläpitäminen ja mahdollinen projektin jatkokehitys olisi helpompaa.

7.4 Projektin vaiheet ja kehitysmalli

Projektissa noudatettiin projektia varten räätälöityä ketterää Petke-kehitysmallia. Alunperin tarkoituksena oli soveltaa jotakin valmista ketterää kehitysmetodia, kuten Extreme Programmingia tai Scrumia, mutta jotkin näiden menetelmien tärkeinä pidetyt ominaisuudet (Extreme Programmingin pari-ohjelmointi, päivittäiset Scrum-kokoukset jne.) eivät tuntuneet soveltuvan kovin hyvin yliopiston tutkimusprojektin ympäristöön ja luonteeseen, joten projektissa päädyttiin yhdistelemään jo olemassaolevia ketteriä menetelmiä sopivalla tavalla.

Projektin aikana pyrittiin seuraamaan ketterää kehitysmallia mahdollisuuksien mukaan. Kehitysmalli toimi projektissa melko hyvin, sillä se pystyi vastaamaan tutkimusprojektissa käytetyn, vielä epäkypsän teknologian asettamiin haasteisiin, jotka ilmenivät muun muassa teknisinä ongelmina sekä aikataulun venymisenä. Lisäksi projektin kantava idea, vuorovaikutuksen tutkiminen, vaati projektilta paljon joustavuutta.

Projektin aikataulu määräytyi kurssin rajaamaan aikaväliin. Aikataulua ei voitu alussa suunnitella kunnolla, koska aihe oli epäselvä, eikä selvitykseen kuluvaa aikaa voinut kunnolla arvioida. Tähän liittyi ongelma-alueen tuntemattomuus, sekä ratkaistavan ongelman puute. Loppujen lopuksi aikataulu jakautui kahteen osaan: selvitysvaihe sekä iteraatiovaihe. Projektin keskeiset tapahtumat on esitelty taulukossa Taulukko 38.

Table 38: Projektin vaiheet

Päivämäärä	Kuvaus
11.10.2007	Projektin aloitus
2.11.2007	Esitutkimuksen tarkastus ja katselmointi
23.11.2007	Projektisuunnitelman katselmointi
10.12.2007	Skenaarioiden palautus asiakkaalle
12.12.2007	Prototyypin esitys Saila Ovaskan UI-kurssilla
21.01.2008	1. iteraation aloituspäivä (Suunnitteluvaihe)
22.01.2008	Asiakastapaaminen, jossa valittiin toteutettava skenaario, Konferenssikani
15.02.2008	1. iteraation lopetuspäivä
18.02.2008	2. iteraation aloituspäivä (Toteutusvaihe I)
07.03.2008	2. iteraation lopetuspäivä
10.03.2008	3. iteraation aloituspäivä (Toteutusvaihe II)
19.03.2008	Määrittelyjen katselmointi
28.03.2008	3. iteraation lopetuspäivä
31.03.2008	4. iteraation aloituspäivä (Testausvaihe)
16.04.2008	Konferenssikani-järjestelmän esittely TKT:n laitoksen tutkijoille
18.04.2008	4. iteraation lopetuspäivä
21.04.2008	5. iteraation aloituspäivä (Projektin lopetusvaihe)
22.04.2008	Käytettävyydestaus
02.05.2008	Konferenssikanin jäädytys versioon 1.0
06.05.2008	Testaussuunnitelman auditointi ja viimeinen tapaaminen asiakkaan kanssa
07.05.2008	5. iteraation ja kurssin lopetuspäivä

7.5 Johtopäätökset

Projekti oli mielenkiintoinen, monipuolinen ja haastava. Kokonaisuutena sitä voidaan myös pitää onnistuneena. Projektin alkuperäisinä tavoitteina oli tutkia vuorovaikutusta julkisessa ja puolijulkisessa tilassa luomalla prototyyppisovelluksia tätä varten. Vaikka prototyyppisovelluksia ei ehditty tehdä enempää kuin yksi, saavutettiin projektissa siitä huolimatta paljon, sillä projektin aikana onnistuttiin toteuttamaan toimiva järjestelmä kaikista kohdatuista haasteista huolimatta.

Vaikka projekti sujuikin hyvin, on aina asioita, jotka voisi tehdä paremmin. Teknologisesta näkökulmasta katsottuna yksi asia, mikä varmasti helpottaisi vastaavanlaista projektia olisi se, että projektissa käytettävä teknologia

olisi kypsempää ja joustavampaa. Näin aikaa ei kuluisi turhaan teknisten ongelmien kanssa painimiseen ja projektissa päästäisiin keskittymään kunnolla kehitettävään järjestelmään. Myös vaatimukset voisivat olla tarkemmat. Dokumentit ja suunnitelmat voisi pyrkiä toimittamaan, mikäli mahdollista, asiakkaalle aikaisemmin jolloin asiakkaalla olisi enemmän aikaa käydä niitä läpi ja antaa kommentteja. Olisi myös hyvä, jos testaukselle varattaisiin enemmän aikaa.

Ryhmätyöskentelymenetelmissä on myös vielä parantamisen varaa. Esimerkiksi palautetta voisi antaa heti kun siihen näkee tarvetta. Lisäksi kommunikointimenetelmiä voisi vielä kehittää. Sähköpostilista on, vaikkakin kohtuullisen toimiva, hidas tapa ryhmän sisäiseen viestintään. Kommunikointiin käytettiin myös IRC-kanavaa, mutta tätä parempi vaihtoehto olisi kenties ollut esimerkiksi jokin pikaviestintäohjelma tai Skype.

Kaiken kaikkiaan projekti on ollut varmastikin kaikille sen jäsenille hyödyllinen kokemus. Vaikka se on vaatinutkin ryhmän jäseniltä paljon energiaa ja pitkäjänniteisyyttä, on siitä ollut myös hyötyä. Jäsenet ovat saaneet lisää sekä teknistä kokemusta että projektityökokemusta ja heillä on ollut mahdollisuus parantaa ryhmätyö- ja vuorovaikutustaitoja. Myös johtamisen kannalta projekti on opettanut paljon käytännön johtamiseen liittyviä isoja ja pieniä asioita.

7.6 Statistiikkaa

Alla on esitetty statistiikkaa projektista.

Table 39: Yleistä tietoa projektista

Ryhmän koko	Kehitys malli	Aloituspäivä	Lopetuspäivä	Päivät	Tunnit	Tunnit / (Päivät * Ryhmän koko)
2+5+2	AGILE	11.10.2007	07.05.2008	210	1240	0.65

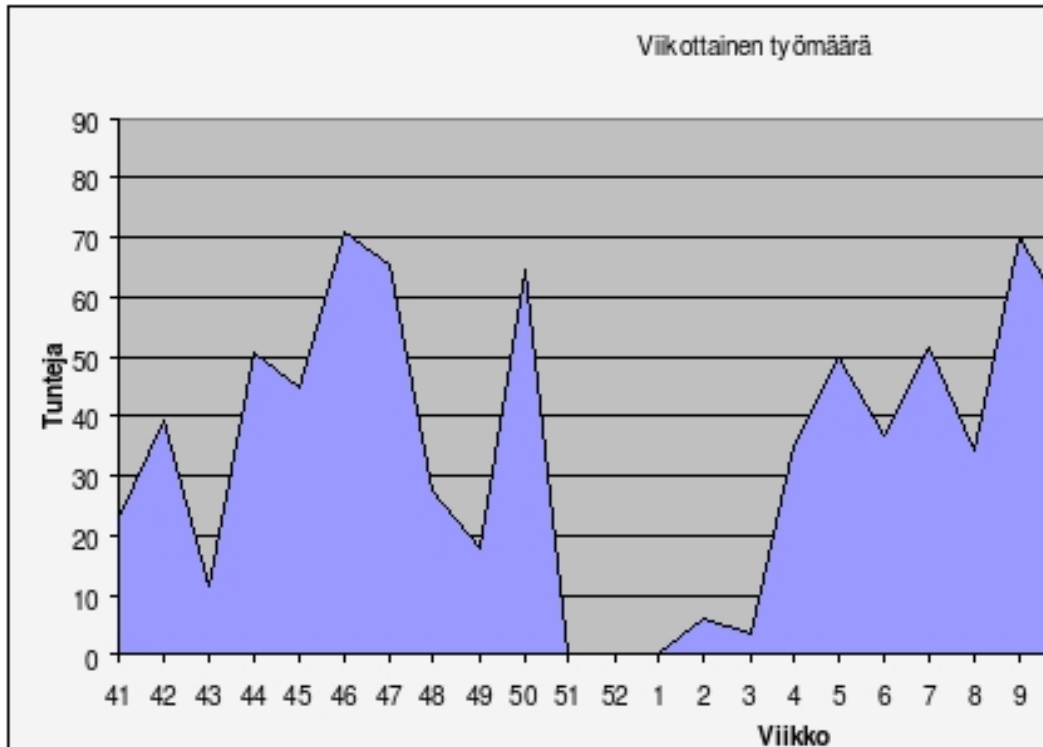


Figure 28: Viikkotunnit

Table 40: Projektiryhmän työtunnit toiminnan mukaan jaoteltuina.

Toiminta	Suun. ja hall.	Vaat. määr.	Design	Koodi	Integ. ja test.	Katselmoinnit	Korjaus	Opiskelu	Muu	Yhteensä
Tunnit	256	70.5	120	189.5	75.5	56.5	42.5	62	154.75	83
%	21	6	10	15	6	5	3	5	12	1027.25
Käytettävyys %										213
Yhteensä										1240.25

Table 41: Vaatimukset ja korkean tason suunnittelun tulokset

Sivut	Vaatimukset	Käyttötapaukset	UI-näytöt	Tietokanta- diagrammit	Tietokanta- taulut
14	63	5		1	10

Table 42: Project's documents.

Dokumentti	sivut	versiot
Preliminäärianalyysi	10	02
Projektisuunnitelma	39	14
Vaatimusmäärittely	14	12
Käyttöliittymä-määrittely	26	09
Kevyt käytettävyyskartoitus	17	07
Käytettävyystestaussuunnitelma	12	10
Käytettävyystestausraportti	26	05
Objektitestausraportti	14	04
Tekninen määrittely	17	04
Testaussuunnitelma	27	08
Skenaario-ideat	11	02
Iteraatioseuranta	13	03
Käyttöohjeet	14	02
Loppuraportti	30	01
Projektitarina	7	01
Viikkoraportit	26	
Total	300	

Table 43: Projektin koodirivit

Kieli	Koodirivit
Java	780
PHP	3891

8 Laitteistotietokanta

8.1 Yleistä

Laitteistotietokanta projektissa toteutettiin Tampereen yliopiston Tietojenkäsittelytieteiden laitoksen käyttöön järjestelmä laitoksen omien käyttäjätunnusten, laitteiden ja ohjelmistolisenssien hallintaa varten. Järjestelmä on tarkoitettu korvaamaan nykyinen, lähinnä paperityöhön perustunut kirjanpito ja osaltaan auttamaan tietojen pysymisessä ajantasalla ja kaikkien ylläpitäjien saatavilla. Järjestelmä toteutettiin PHP:lla www-ympäristössä toimivaksi.

8.2 Projektioorganisaatio

Projektin asiakkaana oli Tampereen yliopiston Tietojenkäsittelytieteiden laitos. Asiakasedustajina projektissa olivat:

- Jori Mäntysalo
- Tuomas Tauriala

Projektipäälliköinä toimivat:

- Jussi Jokela
- Timo Taipalus

Projektiryhmän jäsenet olivat:

- Silvio Quevedo Vázquez
- Sami Blommendahl
- Petri Molkkari
- Jussi Taskinen
- Saila Koironen (käytettävyysryhmä)

Projektijäsenet (pl. käytettävyysasiantuntijamme) eivät jakautuneet erikseen esimerkiksi koodaajiin ja dokumentoijiin/testaajiin, vaan kaikki osallistuivat niin toteutukseen, kuin dokumentointiin. Käytettävyysasiantuntijamme suunnitteli järjestelmän käyttöliittymän ulkoasun ja auttoi projektin

eri vaiheissa erilaisissa käyttöliittymään ja käytettävyyteen liittyvissä asioissa. Yksi ryhmän jäsen lopetti kevään aikana.

Ryhmän kommunikointi tapahtui pääasiassa ryhmän sähköpostilistan välityksellä ja tapaamisissa, joita ryhmä piti lähes viikoittain. Myös pikaviestimiä käytettiin jonkin verran, mutta yhteisten koodausaikojen puuttuessa niiden merkitys jäi vähäiseksi.

8.3 Käytetyt välineet ja työkalut

Toteutusta aloitettaessa ryhmä sai valita toteutuksessa käytettävät tekniikat, asiakkaan asettamien vaatimusten puitteista, joista tärkein oli järjestelmän www-pohjaisuus. Asiakkaan puolelta ehdotettiin käytettäväksi joko PHP:ta tai Javaa projektin toteutuksessa, joista päädyttiinkin käyttämään PHP:tä. Tämän lisäksi päädyttiin käyttämään Apachea ja PostgreSQL -tietokantaa. Tekniikoiden valintaan vaikuttivatkin asiakkaalla jo käytössä olevat tekniikat ja projektiryhmän tietotaito. Vaadittu ympäristö asennettiin laitoksen tarjoamalla virtuaalipalvelimelle.

Itse kehitystyön jokainen projektin jäsen teki haluamallaan työkalulla, mutta suosituin oli Eclipse varustettuna PHP-tuella ja Subclipseella versionhallintaa varten. Myös Crimson Editorin ja TortoiseSVN:n yhdistelmää käytettiin projektin työkaluina. Versionhallinnan toimintaan saaminen aiheutti projektissa hieman haasteita, mutta saatuamme sen kunnolla käyttöön, se auttoi projektin toteuttamisessa. Projektin testausta varten ja samalla lopukäyttäjille annettavaksi selainsuosituksiksi päätettiin valita Mozilla Firefox selain.

8.4 Projektin vaiheet ja kehitysmalli

Projektin luonteen vuoksi, päätettiin projektissa valita inkrementaalinen kehitysmalli ja jakaa projekti neljään inkrementtiin sen eri ominaisuuksien mukaan. Nämä inkrementit ovat riippuvaisia toisistaan, joten ne on toteutettava järjestyksessä. Valitut inkrementit olivat:

1. Kirjautumistoiminnot ja runko
2. Käyttäjätunnusten hallinta
3. Laitteiden hallinta
4. Ohjelmistolisenssien hallinta (jätettiin projektin ulkopuolelle)

Toteutuksen käynnistämässä oli heti alussa haasteita, jonka vuoksi projekti jäi aikataulusta jälkeen. Myös ajan löytäminen projektille tuntui ryhmäläisistä välillä hankalalta muiden opiskelujen, töiden, ynnä muun lomasta. Keskusteltuamme asiakkaan kanssa, päätimme varmistaa komponenttien valmistumisen kunnolla, luopua viimeisen inkrementin toteutuksesta kokonaan. Jälkikäteen onkin helppo sanoa Timo Porasen kehoituksen aloittaa projektin toteuttaminen mahdollisimman aikaisin, olleen täysin paikallaan. Näin kehitysympäristössä ja kehitystyökaluissa kohdattujen ongelmien ratkaisemiseen olisi jäänyt enemmän aikaa.

8.5 Johtopäätökset

Vaikka projekti ei saavuttanutkaan sille asetettuja tavoitteita, ryhmässä valitsi hyvä henki ja projektin jäsenet paneutuivat tehtäväänsä. Projekti oli opettavainen, niin kokemuksena, kuin käytettyjen tekniikoiden osaltakin.

8.6 Tilastot

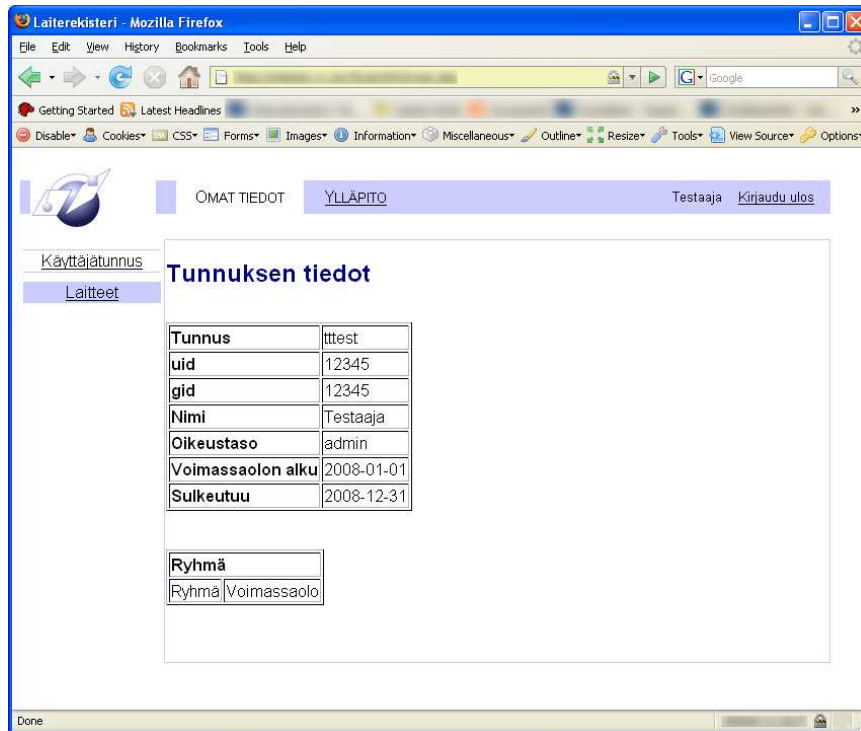


Figure 29: Järjestelmän päänäyttö kirjautumisen jälkeen.

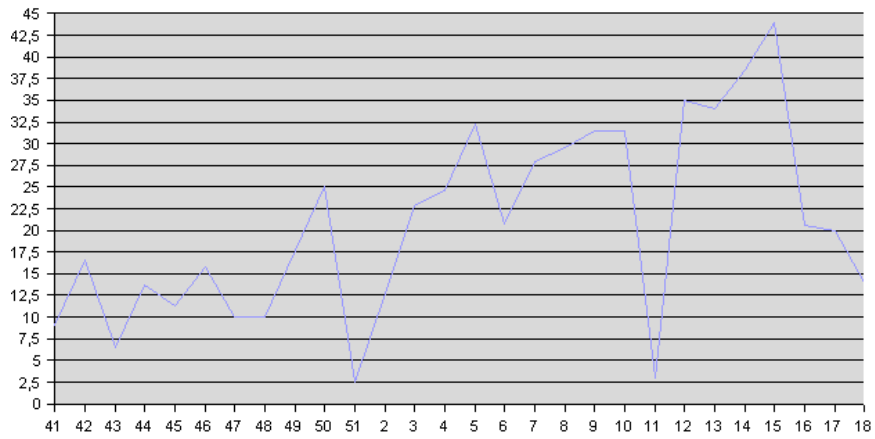


Figure 30: Projektin viikottaiset työtunnit.

Table 44: Yleistietoja projektista.

Ryhmän koko	Kehitys-malli	Tunnit
2+4+1	Inkrementaalinen	600

9 Statua

9.1 Overview

Statua is a GPL-licensed image viewer application for Nokia Internet Tablets. It was meant to replace the original one by being faster, leaner and meaner. It includes features such as slideshow capability, pluggable transition effects and the ability to use Picasa and Flickr as the source for its pictures. It includes also a gesture based navigation where one can draw command to the touchscreen instead of tapping minimalistic icons.

9.2 Organisation and management

- Client
 - Teleca Finland <http://www.teleca.com>
- Project Managers
 - Mikko "*Project Manager*" Kokkonen
 - Wellu "*Primus Motor*" Makinen
- Other Members
 - Jesse "*Energetic*" Laanti
 - Mikko "*Sleepless*" Rantanen
 - Matti "*Loitering*" Ollila
 - Antti "*Knowhow*" Jarvinen
 - Tommi "*Tireless*" Pirttiniemi
 - Jussi "*Usability*" Maaniitty (part time usability expert)

9.3 Methods and tools

- Maemo SDK 4.0 Chinook, <http://www.maemo.org>
- VMWare Player, <http://www.vmware.com>
- Xephyr X server
- Subversion, <http://subversion.tigris.org>
- Eclipse IDE, <http://www.eclipse.org>

- GNU Debug
- Scrum development model, <http://www.controlchaos.com>

9.4 Sample screenshots

Some sample screenshots of our program. There are more screenshots available at our homepage.



Figure 31: Project logo



Figure 32: Application screenshot with overlay controls



Figure 33: Application in thumbnail view

9.5 Homepage

Our project had its very own homepage and wiki at <http://www.statua.org> and <http://wiki.statua.org>. Source code and such are also available at Maemo Garage at <https://garage.maemo.org/projects/statua>.

9.6 Project phases and development model

Project used agile development method so the deadlines weren't as clearly planned beforehand. Instead we planned one week ahead what should have done. In the big schedules we had few dates when we should have product done and when the testing should been started but because of the problems we encountered during development this schedule did not worked well on our project and the testing part had to leave outside.

9.7 Conclusions

Project provided good insight into Linux software developement for mobile devices. The project was pretty much characterized by learning new things. Nokia Internet table as device, Linux as developement environment, Maemo as platform and C as programming language were more or less new to all project members. Project also provided members with knowledge about auxiliary subjects like Subversion version control and Scrum as developement model.

Although project only achieved to produce very basic functionality for the image viewer application the project members gained precious information about mobile development. Even though it is very unlike that Statua will be further developed project members can use their information for the benefit of other mobile projects.

9.8 Statistics

Note that our documenting tool was Wiki instead of static documents thus versions and page numbers are more or less guestimates. Documents were edited all the time and Wiki contained always the latest and correct one that we used during the project.

Table 45: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
2+5+1	Scrum	12.10.2007	29.5.2008	230	938.75	0.510190217

Table 46: Group effort by activity

Activity	Plan. and man.	Req. spec.	Des-ign	Code	Integ. and test	Rev-iews	Re-pair	Study	Other	Total
Hours	242.25	48	27.5	281.5	6	18	96.5	97	816.75	
%	30	6	3	34	1	2	0	12	12	100
Usability										122
Total										938.75

Table 47: Project documentation

Document	Pages	versions
Preliminary analysis	4	2
Project plan	30	4
Project's usability plan	10	3
Requirements specification	1	10
Design plan	1	1
User interface document	2	4
Test plan	0	0
User's guide	1	2
Test report	0	0
Gestures	1	2
Usability test report	0	0
UI Heuristic evaluation report	0	0
Final report	21	1
Final story	4	2
Weekly reports	20+	
Inspection reports	5	
Total	80-100	

Table 48: Productivity metrics.

PM	LOC / PM	Classes / PM	RS & DES pages / PM	LOC pages / PM	Total pages / PM
	1991	N/A	?	39	39 + ?

9.9 Codelines

As a whole, our software consists of almost 1400 lines of code (-the exact line count is 1368, comment lines included). This total amount is divided so that ImageLibrary, the part of the program that handles handles the actual image files, takes up about 530 lines. UI, the part visible to the user and the controls, take up some 340 lines. The Utils section (which contains gesture handling functions and functions for handling resizing and scaling images as well as some functions for handling external image sources) and the files contained in the top directory (which include the main program, the makefile and the appdata file) take up approximately 500 lines of code.

Table 49: Codelines

Directory	Type of code	Amount of code lines
imagelibrary	ANSI C	527
ui	ANSI C	347
utils	ANSI C	336
top	ANSI C	158

10 Scifi

10.1 Yleistä

Projektin tavoitteena oli toteuttaa verkkokauppa scifielokuvien harrastajille. Toteutettu lopputuote oli sivustokokonaisuus, johon projektissa toteutettiin elokuvien esittelyn, lisäämisen, ostamisen, hakemisen ja verkkokaupan lisäksi blogi-toiminto, käyttäjienhallinta, navigoinnihallinta, uutiset, bannerit, kuukauden motto, kävijälaskuri, vinkkaa kaverille ja elokuvien kommentointi ja arvostelu. Sivusto pohjautuu aktiiviseen yhteisöön ja käyttäjiin, jotka tarjoavat omia elokuviansa sivustolle. Sivuston ympärille on tarkoitus tehdä yritystoimintaa. Projektin lopputuotetta ei julkaista sellaisenaan, vaan asiakas muokkaa tuotetta lopullisen käyttötarkoituksen mukaan. Projektin toimeksiantajana toimi yritys ADA Oy.

Projektin kotisivu löytyy osoitteesta : <http://www.cs.uta.fi/~scifi/>.

10.2 Projektiryhmän jäsenet

Projektipäällikköinä toimivat Tomi Saarinen ja Mika Ekman. Muita jäseniä olivat Mikko Viskari, Juuso Kosonen, Tommi Kivistö ja Anssi Männistö. Käytettyvyystiimissä (UI) toimivat Johanna Huhtala ja Arttu Ekholm.

Seuraavassa jäsenten vastuut eriteltynä tehtävien mukaan:

Table 50: Jäsenten vastuut.

Tehtävä	Vastuussa
Projektin johtaminen	Tomi Saarinen
Esitutkimus	Koko ryhmä
Vaatimusmäärittely	Koko ryhmä
Testaussuunnitelma	Koko ryhmä
Toteutussuunnitelma	Koko ryhmä
Toteutus	Mikko Viskari, Juuso Kosonen, Anssi Männistö ja Mika Ekman
Testaus	Anssi Männistö
Käyttöliittymä	Johanna Huhtala, Arttu Ekholm
Loppuraportti ja tiivistelmä	Tommi Kivistö



Figure 34: Projektin jäsenet

10.3 Käytetyt välineet

Www-sivusto toteutettiin pääosin PHP-ohjelmointikielellä ja lisäksi käytettiin Javascriptiä osaan sivuston toiminnallisuuksia. Muita käytettyjä ohjelmistoja olivat:

- Eclipse ohjelmointiympäristö
- Subclipse versionhallinta
- MySQL tietokanta
- XAMPP palvelinohjelmistopaketti
- Microsoft Office Word tekstinkäsittely
- StarUML mallinnusohjelma

Kaikkea ei ohjelmoitu alusta asti itse, vaan noin puolet koodiriveistä (SLOC) hyödynnettiin valmiita komponentteja, kuten Javascriptillä tehty kalenteri.

10.4 Projektin vaiheet

Projekti aloitettiin syksyllä 17.10.2007 ensimmäisellä viikkopalaverilla, joita pidettiin kerran viikossa koko projektin ajan. Asiakas tavattiin ensimmäisen kerran lokakuussa 2007. Projektin aiheen hahmottuessa luotiin projektisuunnitelma ja aikataulut.

Projektin kehitysmallina toimi inkrementaalinen ohjelmistokehitys. Scifi-projekti jaettiin neljään inkrementtiin. Projektin ensimmäinen inkrementti osoittautui työläämmäksi mitä oli projektin alussa suunniteltu ja koko projektin aikataulusuunnitelmaa jouduttiin hieman korjaamaan kaikkien inkrementtien osalta siten, että kokonaisuus kuitenkin pysyi aikataulussa. Kaikkiaan kuitenkin projektin voidaan sanoa pysyneen erinomaisesti aikataulusaan.

Projektiryhmä jaettiin siten, että toisen projektipäällikön ja osan ryhmäläisistä vastuulle jäi toteutus ja testaus. Toisen projektipäällikön ja osalle ryhmästä vastuulle jäi vaatimus-, toteutus- ja testaussuunnitelmat ja muut dokumentit.

Projekti saatiin päätökseen 7.5.2008.

10.5 Johtopäätökset

Projektiryhmä sai kurssin aikana tutustua ohjelmistokehityksen koko elinkaareen ja projektin hallintaan. Projektin vaiheissa tuli tutuksi projektinhallinta, aikataulutus, dokumentointi, suunnittelu, testaus ja versionhallinta.

10.6 Tilastoja

Table 51: Yleisinformaatiota.

Koko	Keh. malli	Aloituspvm.	Lopetus pvm.	Päivät	Tunnit	Tunnit/ (Päivät*Ryhmän koko)
2+4	INC	17.10.2007	7.5.2008	196	1008	0,86

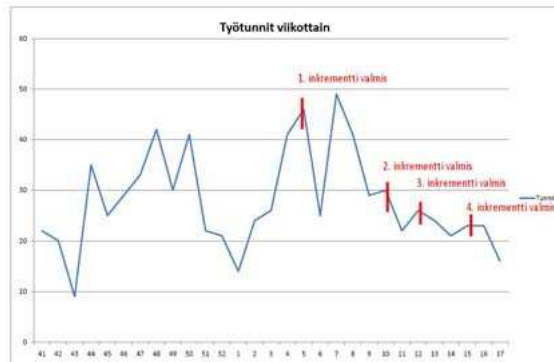


Figure 35: Viikottaiset tuntimäärät.

Table 52: Tuntimäärät työlajeittain.

Työlaji	Joh- tami- nen	suun- nitte- lu	Vaati- määr.	Ohjel- mointi	Inte- groidi	Katsel- moinnit	Muut	UI	Yhteensä
Tunnit	10	32	3	22	0	6	12	13	100,0
%									
Yht.	103	316	31	218	18	67	124	131	1008

Table 53: Projektin dokumentit.

Dokumentti	Sivuja	Versioita
Esitutkimus	14	3
Projektisuunnitelma	31	6
Vaatimusmäärittely	21	9
Toteutussuunnitelma	24	8
Testaussuunnitelma	24	8
Testiraportit	19	1
Käytettävyyskartoitus	8	2
Käyttöliittymäsuunnitelma	19	3
Käytettävyden arviointi	9	3
Navigoinnin hallinta	4	1
Asennusohje	4	2
Loppuraportti	26	3
Yhteensä 48	204	

Table 54: Projektin koodirivit.

Kieli	PHP, JS, HTML
LOC	44307
SLOC	35345
Uudelleen käytettyä koodia	23327
Uudelleen käytettyä koodia (%)	66

Table 55: Tuottavuus metriikat suhteutettuna projektikuukausiin.

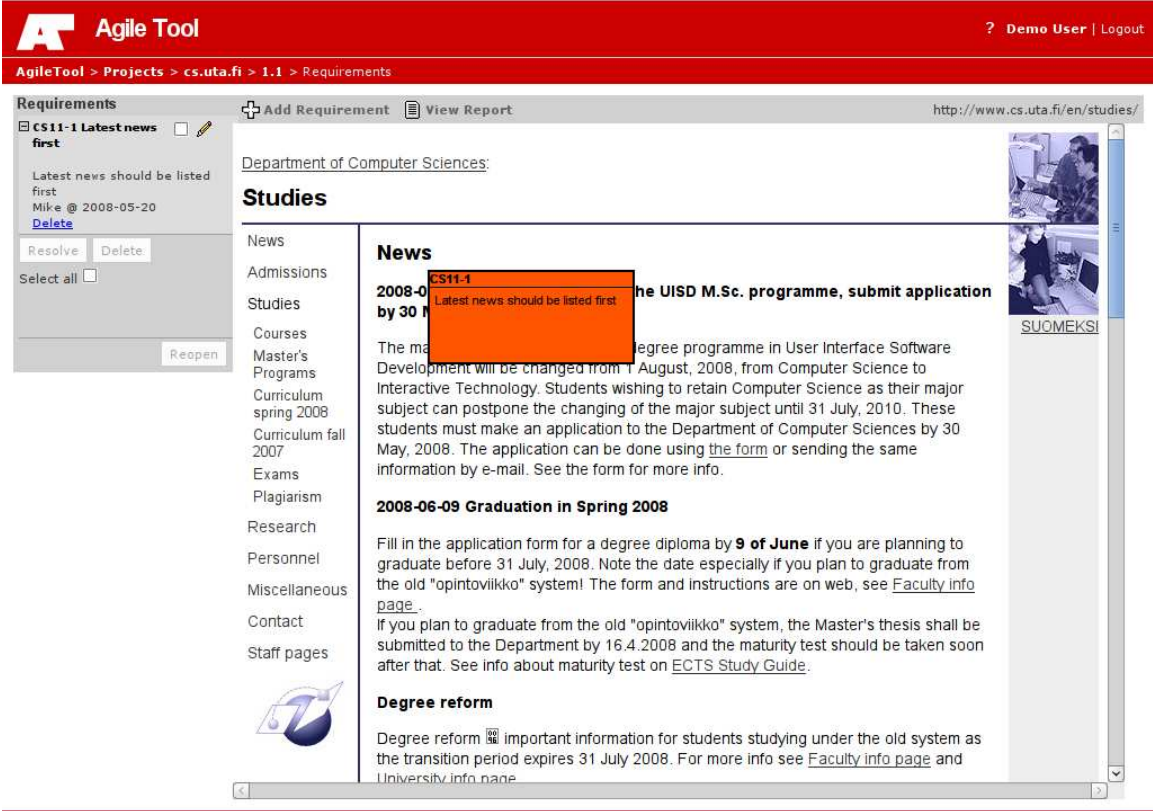
PK	LOC / PK	Luokat / PK	VM & TS sivut / PK	LOC sivut / PK	Sivut yht. / PK
8	5538	33,6	5,6	73,8	25,5

11 AgileTool

11.1 Yleiskuvaus projektista

11.1.1 Tuote ja ympäristö

Projektin tarkoituksena oli kehittää WWW-pohjainen työkalu, joka auttaa hallitsemaan ja ylläpitämään vaatimuksia ketterien menetelmien (Agile Methods) ohjelmistoprojekteissa. Kehitettävän sovelluksen tarkoitus on helpottaa vaatimusten kirjaamista ja parantaa niiden jäljitettävyyttä lähinnä WWW-pohjaisia sovelluksia toteutettaessa. Ohjelmisto mahdollistaa projektin vaatimusten liittämisen suoraan kehitettävän sovelluksen toimiviin näkymiin. Sovellus asennetaan yhdelle palvelimelle, josta sitä voidaan käyttää keskitetysti useamman projektin hallinnoimiseen.



The screenshot displays the AgileTool web interface. At the top, there is a red header with the Agile Tool logo and navigation links like "Demo User" and "Logout". Below the header, a breadcrumb trail shows the current location: "AgileTool > Projects > cs.uta.fi > 1.1 > Requirements". The main content area is divided into several sections. On the left, there is a "Requirements" sidebar with a list of items, including "CS11-1 Latest news first". The main content area features a "Studies" section with a "News" sub-section. A red box highlights a specific news item: "CS11-1 Latest news should be listed first". Below this, there are sections for "2008-09 Graduation in Spring 2008" and "Degree reform". The interface also includes a navigation menu on the left with categories like "Admissions", "Courses", and "Exams".

Figure 36: AgileToolin vaatimusnäkyvä.

11.1.2 Ohjelman toimiminen osana järjestelmää

Ohjelmisto toimii www:n yli palvelimelta, jonne on asennettuna Grails -ympäristö. Käytännössä AgileTool on kiinteä osa asiakasyrityksen projektien läpiviemistä ja www-pohjaisten projektien iterointia asiakkaan kanssa. Toteutuksen pohjana on nykyinen toimintamalli ja tämän mallin tukeminen mahdollisimman kattavasti.

11.1.3 Rajallisuudet ja rajoitteet

Ohjelmaa toteutettaessa pyrittiin siihen, että minkä tahansa www-sivuston tai palvelun voi ottaa käsiteltäväksi AgileTool -järjestelmään. AgileTool ei siis suoraan tue muita kuin selainpohjaisia projekteja. Järjestelmä ei ole selainriippuvainen, mutta vaatii joidenkin perustekniikoiden käyttöä selaimessa. Työkalussa on joitakin rajoituksia, mutta toimii hyvin asiakasorganisaation projektien puitteissa. Tuotteen rajoituksia on mietitty tarkemmin jatkokehitysideoissa.

11.2 Projektiorganisaatio

AgileTool -projektiryhmä (kuva 37) koostui yhteensä yhdeksästä ihmisestä. Ryhmän projektipäällikköinä toimivat Mike Arvela ja Teppo Tomann. Ryhmän jäseniä olivat Juho Hella, Jukka Hell, Jukka Pollari, Oskari Salonen ja Petri Kotiranta. Lisäksi käytettävyyssryhmästä tulivat Juha Pakkanen ja Saira Koiranen.



Figure 37: AgileTool-projektiryhmä: JukkaP, Juho, Juha, Mike, Petri, Teppo ja Oskari.

11.3 Työskentelymetodit ja työkalut

Projektin kehitysmalli perustuu ketterään Scrum-projektimalliin. Kaikkiaan projektissa pyritään parhaimman mukaan noudattamaan seuraavia Scrumin määrittelemiä tai sille ominaisia käytäntöjä:

- vaatimusmäärittelydokumentin korvaaminen tuotteen kaikki osa-alueet kattavalla backlog-listalla (Product Backlog)
- työn jakaminen kiinteään mittaisiin ajanjaksoihin, sprintteihin (Sprint Planning) valitsemalla backlogista tietty kokonaisuus toteutettavaksi (Selected Product Backlog)
- ryhmäläisten työn säännöllinen seuraaminen (Daily Sprint): normaalisti toteutettu päivittäin suullisesti, projektissa olosuhteiden vuoksi Skype:n avulla
- sprintin työstäminen loppuun asti atomisesti (ei muutoksia ennen seuraavaa sprinttiä)
- katselmointi kunkin sprintin päätteeksi (Sprint Review), jolloin asiakkaalla mahdollisuus vaikuttaa seuraavan sprintin sisältöön
- sprintin päättyessä keskustelu sen onnistumisesta ryhmäläisten kanssa (Reflection): mitä hyvää, mitä huonoa, mitä tehtäisiin eri tavalla.

Projektin toteutuksessa käytettiin Grails-kehiksen vakainta saatavilla olevaa versiota. Kehitysvälineinä käytettiin Grailsin tarjoamia komentorivityökaluja sekä graafisena kehitysympäristönä Eclipseä, johon otettiin käyttöön tarvittavia lisäominaisuuksia erilaisilla kolmansien osapuolten plugineilla. Osa ryhmäläisistä käytti myös IntelliJ IDEA -kehitysympäristöä Eclipsen sijaan kehitysympäristönä. Grails-kehys sisältää itsessään monia valmiita komponentteja kuten Hibernaten tietokantayhteyksiä varten, Jetty-sovelluspalvelimen ja Log4J-lokittajan, joita hyödynnettiin tarpeen mukaan kehitystyössä. Sovelluksen referenssiselaimena käytettiin Internet Explorer 6:sta. Tämän lisäksi järjestelmän toimivuutta testattiin Firefox 2:lla, Operalla ja Internet Explorer 7:llä. Versionhallinnassa käytettiin SVN:ää.

Projektin kuluessa ryhmäläiset tulivat käyttäneeksi mm. seuraavia ohjelmia ja työkaluja: Eclipse, IntelliJ IDEA, Mozilla Firefox, Opera, IE 7, GanttProject, EditPad, Windows XP, OpenOffice, Skype, TortoiseSVN, Subversion ja Grails. Lisäksi työtuntien merkitsemiseen ja ylläpitoon käytettiin Matias Muhosen viime vuonna projektityökurssia varten tekemää Timelet-sovellusta (<http://timelet.googlecode.com/>).

11.4 Projektin vaiheet

Projektissa oli kymmenen työvaihetta, sprinttiä (Scrum), jotka kestivät projektin vaiheen mukaan pituudeltaan tavallisesti kaksi tai kolme viikkoa. Projektin alussa panostettiin lyhyempiin sprintteihin, jotta toteutettavan työkalun kehitys saatiin ohjattua paremmin oikeaan suuntaan. Sprinttien sisältö pohjautui Product Backlogiin, josta valittiin asiakkaan kanssa kussakin sprintissä toteutettavat osa-alueet.

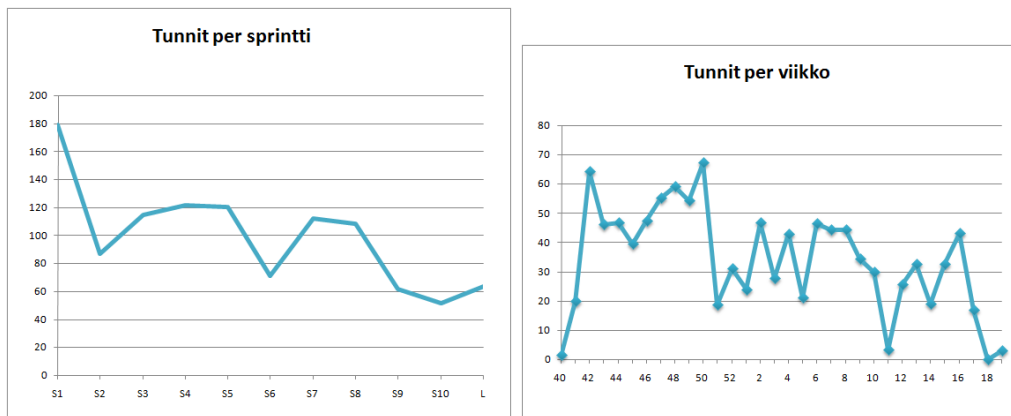


Figure 38: Ryhmän työtunnit sprinteittäin ja viikoittain.

Kaksi ensimmäistä sprinttiä olivat pääasiassa suunnittelua ja projektin käynnistelyä, ja sprintistä 3 eteenpäin siirryttiin varsinaiseen toteutukseen. Sprintti 5 ajoittui joulun lomien yli, minkä takia sen pituus oli tarkoituksellisesti muita pidempi. Sprintti 6 palasi jälleen tiiviimpään aikatauluun, jotta päästiin pyhien jälkeen paremmin käsiksi itse projektiin.

Sprinteille suunnitellut sisällöt tarkentuivat toteutuksen edetessä. Sprintin sisällön painopiste määräytyi aina edellisen sprintin katselmoinnissa asiakkaan toivomusten ja projektiryhmän ehdottamien osa-alueiden mukaan.

11.5 Yhteenveto

Projekti oli varsin opettavainen. Kurssin aikana sai paremmin näkemystä siitä, millainen ohjelmistoprojekti käytännössä on (kuka tekee mitään, vastualueet, miten projekti etenee käytännössä). Lisäksi oli varsin mielenkiintoista saada tuntumaa yrityksen näkökulmaan. Projektin aikana tuli opittua uusia tekniikoita, kuten GRAILS -ympäristö ja tutustuttua erilaisiin ohjelmistokehityksen ohjelmistoihin, kuten Eclipseen ja versionhallintaan. Ryhmätyöskentely ohjelmistoprojektissa oli uutta ja mielenkiintoista.

Table 56: Sprinttien aikataulus ja sisältö.

Sprintti (kesto)	Ajankohta	Sisältö
1	- 2.11.	HTML-mallin ensimmäinen versio
2	5.11. - 16.11.	HTML-mallin toinen versio
3 (2vk)	19.11. - 30.11.	HTML-mallin siirtäminen Grails-kehiksen päälle alkaa
4 (2vk)	3.12. - 14.12.	Mallin siirtäminen Grailsille jatkuu, pääpaino vaatimusten liittämässä näkyymiin
5 (4vk)	17.12. - 11.1.	Proxy, vaatimusten käsittely, raportointinäkyvä, käyttöliittymän suunnittelu
6 (2vk)	14.1. - 25.1.	Vaatimusten liittäminen, käyttöliittymän toteuttaminen, syötteiden esittäminen, kirjautumistoiminnot
7 (3vk)	28.1. - 15.2.	Käyttöliittymä, kirjautumistoiminnot
8 (3vk)	18.2. - 7.3.	Raportointiominaisuudet (vaatimuslistan tulostaminen)
9 (3vk)	10.3. - 28.3.	Testausta ja viimeistelyä
10 (2vk)	31.3. - 15.4.	Viimeistelyä

Projekti onnistui ryhmän osalta hyvin. Tämä johtui muun muassa siitä, että kaikki tekivät hommansa hyvin ja olivat kiinnostuneita projektista. Toteutettu sovellus on hyvä, varsinkin kun ajattelee projektiin käytetyn ajan määrää. Isommalla työmäärällä olisi saatu aikaan vielä parempi sovellus, mutta valitettavasti projektin aikataulu ei antanut tähän mahdollisuutta. Tarvittaessa asiakkaamme Ambientia tulee korjaamaan niitä puutteita, joita ohjelmaan vielä jäi. Tällä hetkellä Ambientia vaikuttaa kuitenkin erittäin tyytyväiseltä tuotteeseen ja tulee ottamaan sen käyttöön omilla projekteissaan nopealla aikataululla. Mikään ei projektissa mennyt pieleen tai epäonnistunut.

Grails-kehiksen valinta ohjelmointityökaluksi oli hyvä ratkaisu. Grailsia tullaan varmasti jatkossakin hyödyntämään muissa projekteissa. Scrum-projektimalli sopi hyvin kyseessä olleeseen projektiin ja ryhmän tarpeisiin. Projektimallista saatu kokemus oli erittäin arvokasta.

11.6 Tilastoja

Projektissa käytettyjen Scrum-projektimallin ja Grails-kehityksen seurauksena sen tunnusluvut ja tilastot eroavat melko paljon perinteisestä vesiputousmallilla tehdystä projektista. Koska projektimalli korostaa jatkuvasti muuttuvia vaatimuksia ja toisaalta toimivan tuotteen tärkeyttä dokumentaatioon nähden, pääpaino dokumentaatiossa oli kurssin kannalta vaadituissa deliveraabeleissa.

Grails-sovellusten luonteen vuoksi generoidun ja itse kirjoitetun koodin välille on melko vaikeaa tehdä tarkkaa eroa, sillä jo tyhjä Grails-projekti on itsessään valmis Java EE -sovellus. Laskentaa hankaloittaa myös verrattain uuden Groovy-kielen puuttuminen laskentaohjelmistojen, kuten suosittuun sloccountin (<http://www.dwheeler.com/sloccount/>) tukemien kielten listalta.

Table 57: Yleistietoa projektista.

Ryhmä- koko	Keh. malli	Aloitus pvm	Lopetus pvm	Päivät	Tunnit	Tunnit / (Päivät*Ryhmä)
5+2+2	Scrum	4.10.2008	18.5.2008	228	1090	0.53

Table 58: Työtunnit työlajeittain.

Kate- go- ria	Proj. suun. ja joht.	Vaat- im- määr.	Suun- nit- telu	Tot. teg.	In- ja test.	Korj.	Kat- sel- mointi	Opisk.	Muut	Yht.
Tunnit	506,85	11,75	13,5	364	14,25	40	27,75	64,75	47	1089,85
%	46,5	1,1	1,2	33,4	1,3	3,7	2,5	5,9	4,3	100
Käytet- tävyys										192,5
Yht.										1282,45

Table 59: Dokumentaatio.

Dokumentti	Sivuja	Versioita
Esitutkimus	16	1
Projektisuunnitelma	34	4
Jatkokehitysideoita - yleiset vaatimukset	5	1
Loppuraportti	26	1
Projektikertomus	7	1
AgileTool in a nutshell	1	1
AgileTool - managing requirements in agile WWW projects	4	(useita)
AgileTool käyttöohje (online)	10	2
Viikkoraportit	22	
Katselmointiraportit	8	
Yhteensä	159	

Table 60: Esitykset.

Esitys	Sivuja
AgileTool esitys 16.10.2007	11
AgileTool esitys 12.12.2007	11
AgileTool esitys 7.5.2007	12

Table 61: Koodin määrä.

Kieli	Groovy
SLOC (*.groovy)	1240
Sivupohjarivejä (*.gsp)	1518
Tuotettuja tiedostoja	40
JIRA-tehtäviä	76

12 ReaP

12.1 Overview

The purpose of the project ReaP was to further develop a mobile TV reporting application for TeliaSonera. The old system at TeliaSonera was slow to use and required lots of manual work. The reporting application the project created, produces OpenDocument format presentations about the usage of the mobile TV application. It analyses data from two different types of log files and combines the results into a OpenDocument presentation. It creates reports automatically and no manual labor is needed.

12.2 Organisation and management

The project group consisted of one project manager, 5 team members and one usability team representative.

- Project Manager: Sari Miettinen
- Project Member: Jarkko Huotari
- Project Member: Olavi Karppi
- Project Member: Laura Markkanen
- Project Member: Piia Perälä
- Project Member: Matti Pesonen
- Usability Team Member: Santtu Mansikkamaa

The team members did not have strict roles during the project, but instead all team members took part in the document creation. Jarkko, Olavi and Matti were mainly responsible of the coding of the application, where as Laura and Piia were responsible of the testing.

12.3 Methods and tools

- Eclipse was used for Java-programming environment and plugin called Subclipse was used to handle the Subversion system. The coding tools selected were useful and helped the coding process, as Eclipse also has a built-in support for Javadoc which was used to comment the code. Furthermore, Eclipse supports automatic code formatting and this was used to follow the Sun's coding conventions for Java.

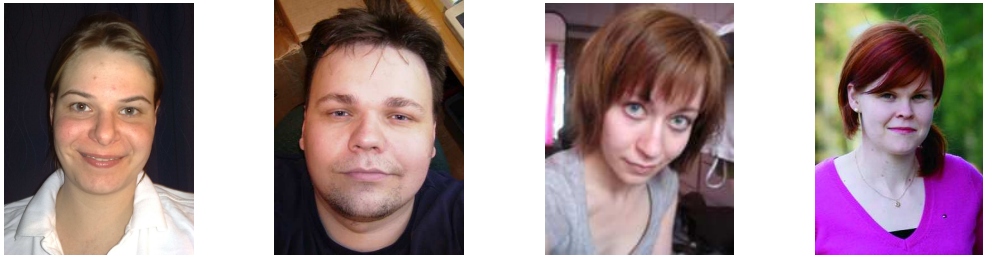


Figure 39: Sari Miettinen, Jarkko Huotari, Laura Markkanen ja Piia Perälä.

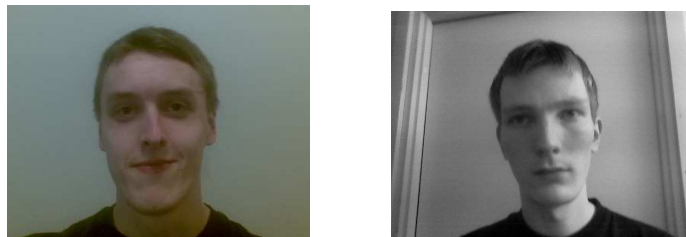


Figure 40: Olavi Karppi and Matti Pesonen.

- Oracle was used as a database during the project. The project team used the Oracle Database 10g Express Edition and it was found interesting to learn new database conventions.
- DokuWiki was used as a wiki-software for project web-pages. The wiki was useful when delivering documents, even though it was found that it could not handle the largest files produced during the project. The wiki was used as a document repository as well as for reporting the hour allocations. Delivering the document via wiki was useful as project members' the email inboxes did not fill up.
- Fujaba, Dia and BoUML was used for modeling. The tools were free-ware and some commercially available tools might have been more agile for project's use.
- Microsoft Word and Open Office was used for document creation. However, it was found that using the both programs was not the best choice, as the formatting of the documents was broken every time the tool was changed.

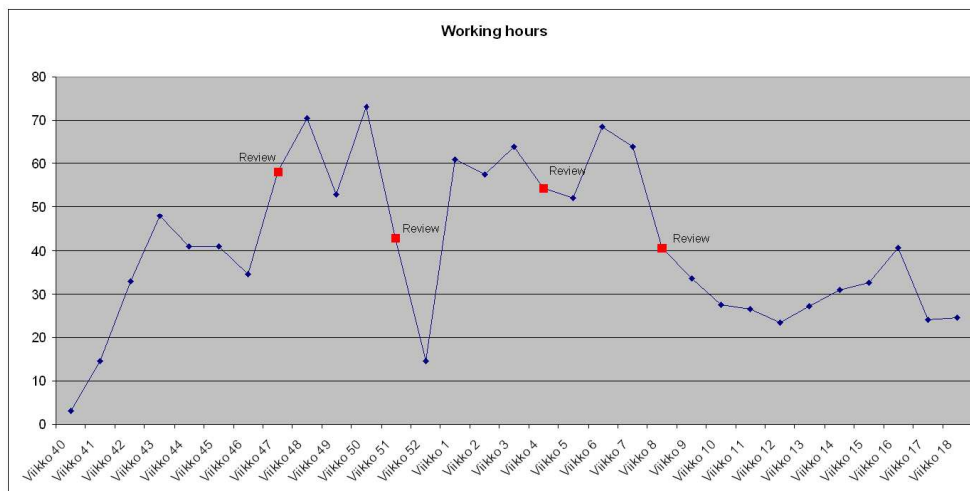


Figure 41: Weekly workload of the project.

12.4 Project phases and development model

The development model for the project was incremental as the project could be clearly divided into three separate phases. The incremental development model was applied as suitable and the increments were overlapping due to time constraints of the project. Incremental development model enabled the coding to begin earlier than in traditional waterfall model and the documentation of the project was more detailed, as the team had to concentrate only to a part of the requirements at a time. The testing of the application proved to be somewhat challenging with incremental development model and time constraints of the project.

12.5 Conclusions

All project members were novices and they did not have any previous experience on working in a real software project. Therefore almost all project members thought that working with the project was very beneficial because the project gave them a chance to learn how working in the software project can be organized.

Project took an academic year therefore members of the project felt that working among the project mixed with other academic courses were quite hard and took too much time. Sometimes it was quite laborious to integrate working with other academic courses' work to the project work.

It could be said that the project could have used more working hours than 7,5hours per week which was the calculated working time for the week

working during the project. There was a great group spirit between project members during project and everybody enjoyed working with each other. Nobody dropped out from the project and everyone worked with a good motivation during the project.

All project members had to write documents during the project. This was quite hard and slow because project's documentation language was English. However, writing documents in English was very good practice for the future work life. Project's topic was interesting and it could be said that it was a great motivator for working among the project.

At the beginning of the project there could have been more code reviews with some experienced coder, because that would be useful for both project's coders and other members. They would get tips for working to the right direction.

In the project quite normal development model was used. Because this was mostly an academic project, it would have been a great opportunity to try a bit more exotic development model, e.g., Scrum.

At the beginning of the project all tools of the project must be chosen very carefully for example tools which are used for modeling and documentations. This can save many working hours during the project, especially if documents are a big part of the project.

It was also found during the project, that the very first testing can be one of the most important testings in the application development. The first testing creates a basic ground to the application.

12.6 Statistics

Table 62: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
1+5+1	INC	2.10.2007	7.5.2008	218	1330	0.87

Table 63: Requirements and high-level design outcome.

Pages	Requirements	Use-cases	UI screens	Database diagrams	Database tables
72	49	0	3	1	9

Table 64: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Des- ign	Code	Integ. and test	Rev- iews	Re- pair	Study	Other	Total
Hours	140	179	73	223	103.5	79.5	29.5	69.5	382.5	1279.5
%	11%	14%	6%	17%	8%	6%	2%	5%	31%	100%
Usability										26.5
Total										1306

Table 65: Design outcome.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
65	1	2	0	3	10

Table 66: Inspection findings.

	Project plan	Increment 1	Increment 2	Increment 3
Pages and/or screens	19	61	64	150
Preparation time	560	780	470	580
Inspection time	999	960	840	969
Findings	110	58	54	68
Used time / findings	3.99	3.10	6.85	5.72

Table 67: Productivity metrics.

PM	LOC / PM	Classes / PM	RS & DES pages / PM	LOC pages / PM	Total pages / PM
8.75	1098.86	16.11	28.8	21.98	56

Table 68: Project's documents.

Document	Pages	versions
Preliminary analysis	16	15
Project plan	21	10
Requirements specification 1	21	19
Design plan 1	15	13
Test plan 1	21	7
Test report 1	10	1
Requirements specification 2	63	11
Design plan 2	25	7
Test plan 2	28	11
Test report 2	5	1
Requirements specification 3	72	4
Slide specification	23	7
Design plan 3	27	3
Test plan 3	27	10
Test report 3	4	1
User interface document	6	10
Maintenance document	38	7
Final report	16	8
Final story	5	3
Weekly reports	32	
Inspection reports	15	
Total	490	

Table 69: Project's codelines.

Language	JAVA	Bourne Shell	SQL	XML	HTML	Total
LOC	8625	48	110	700	132	9615
SLOC	7105	47	110	690	126	8078
Reused code	0	0	0	0	0	0
Reused and modified code	0	0	0	0	0	0
Classes	141	0	0	0	0	141
Functions	483	0	0	0	0	483
Code revisions	432					

13 Gaze Games

13.1 Overview

The goal of the Gaze Games project was to create a platform for implementing gaze-based games with network functionality, and a game that would use this platform. The final deliverables would be two. A library, with its documentation, and a classic battleship game implemented using this library.

13.2 Organisation and management

The client of the project is TAUCHI (Tampere Unit for Computer-Human Interaction), Department of Computer Science of university of Tampere. TAUCHI is represented by Kari-Jouko Rähkä (head of the unit) and Oleg Spakov's also of TAUCHI.

The team consisted of 3 managers and 5 members, with almost everyone coming from a different country. The managers were:

- Hanna Kulas (Poland)
- Spyridon Saroukos (Greece)
- Musa Sizya (Nigeria)

The team members were:

- Cedric Alfonsi (France)
- Nazmie Ali (Turkey)
- Thiago Chaves (Brazil)
- Jevgeni Haigora (Russia)
- Paavo Kilpeläinen (Finland)

The Usability team members were:

- Santtu Mansikkamaa (Finland)
- Marko Oravainen (Finland)



Figure 42: Project members.

13.3 Methods and tools

During the project weekly meetings were arranged, in which open issues were discussed and decisions were taken. In addition the group communicated through email and instant messaging.

The project team used an iterative development model with three iterations. Iterations were less defined than planned but the model was reasonably well suited for the project. A more strict development model such as the waterfall model might have caused the work to be delayed further. Iterative model permitted some on-the-fly rescheduling and rethinking of objectives which helped in finishing critical tasks.

Most of the work during the project was done individually, meetings were used to discuss tasks for the week and possible improvements and new ideas. Testing was mostly done in the gaze lab during/after meetings because of the need for a gaze control device.

- E-mail: Used mainly for reports, reminders and setting meetings by the project managers.
- Eclipse: Main development environment. Used with CDT (version 4) and Cygwin.

- Google Docs: Used extensively in writing project documentation. Working hours and available meeting hours were reported using a spreadsheet in Google Docs. Worked well despite some quirks with different browsers and formatting. Ability to work simultaneously on the same document very useful.
- MSN Instand Messaging: Used for real time communication.
- Subversion: Used for version management of source code.

13.4 Conclusions

Although the platform is not considered mature, the team manage to test it and document it. The library implements all the requirements the client requested and is a good basis for further development. The client and the team members are satisfied with the final result.

13.5 Statistics

The team produced 2 software products of 4700 LOC in total. Of these, 2500 belong to the platform and it's libraries and can be reused in developing other games and the rest 2200 belong to the demo game.

Table 70: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
3+5+2	Incremental	12.10.2006	24.5.2007	226	1086	0.6

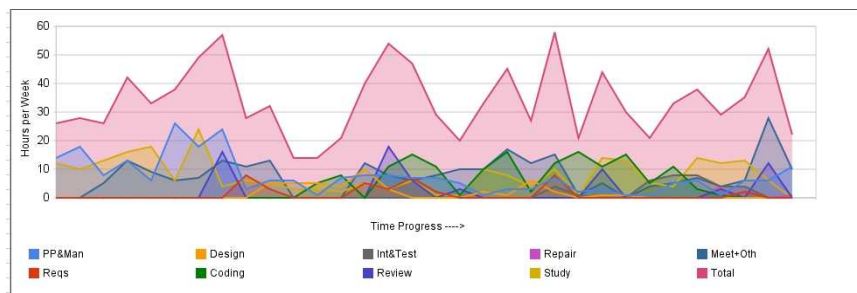


Figure 43: Weekly workload of the project per category

Table 71: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Design	Code	Integ. and test	Reviews	Re-pair	Study	Other	Total
Hours	241	38	45	153	35	68	9	252	245	1086
%	21	3	4	14	3	6	1	24	23	
Total										

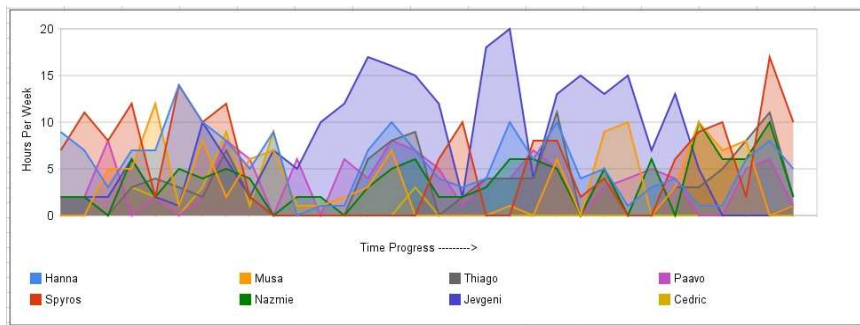


Figure 44: Weekly workload of the project per team member

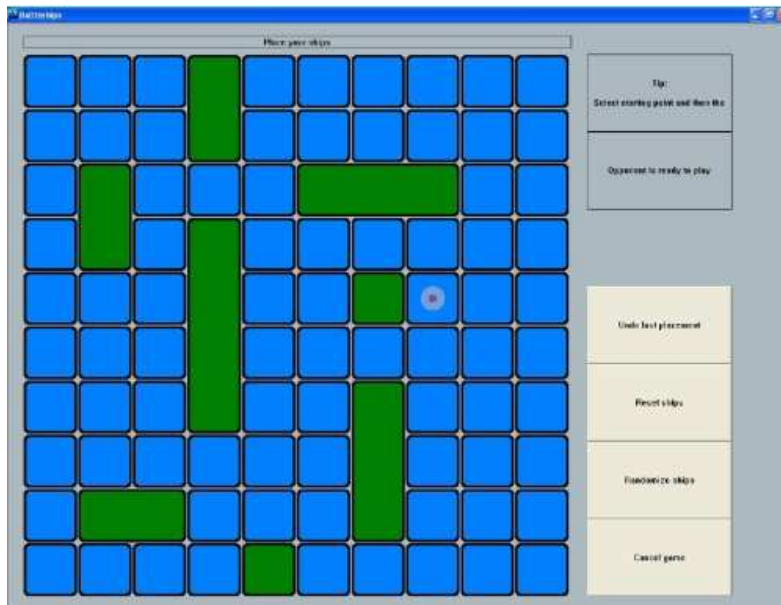


Figure 45: Screenshot of the battleships game implemented with the platform.



Figure 46: Testing session in Fysio Lab.

14 RVLE

14.1 Overview

The purpose of this project was to resurrect an existing virtual learning environment system (VROMOSS) for the project's client, TAUCHI. A functional system needed to be set up on the client's hardware and parts of the system software needed be modified and developed further. The development work entailed at least the implementation of an open source 3D library for the 3D displays and adding support for modern haptic feedback control devices. However, because the existing system was built almost ten year ago and we fail to solve compatibility problem of related libraries, these project suffer a failure at the end.

The address of our project is: <http://www.cs.uta.fi/~rv79775/rvle/>.



Figure 47: VROMOSS system screenshot.

14.2 Organisation and management

The project group consisted of 7 students of the University of Tampere at the beginning, but two student quited the lecture in the middle, so there are actually 5 student who actually contribute to the project. There are 2 project managers who are managing the project as a part of their software project management course, 3 group members from the project work course and 1 member from the usability group of the same course.

Two project managers Peng Wang and Suprasanna Sarkar divided their job, so that Peng Wang was project chief manager in autumn 2007 and Suprasanna was project chief manager in spring 2008. The main responsibility for the chief manager is to arrange weekly project member meeting, client meeting, review meeting, writing weekly report, establish detail project plan, assigning tasks and other managerment stuff, the vice manager acts as a assistant who help the chief manager to fulfill his responsibility.

- Project Manager: Peng Wang, Suprasanna Sarkar
- Web site developer and maintainer: Risto Välimäki
- System developer: Jussi-Antti Salomaa, Risto Vïimäki, Niran Malakar
- Usability officer (Usability team): Riki Kawakami



Figure 48: Project members.

14.3 Methods and tools

The tools used in our project are listed below.

- Microsoft Visual Studio 2005. Purpose: Using to build and run the solar system C++ code.

- Eclipse SDK 3.2. Purpose: Using to build and run the Java mission server component.
- Flash plug-in and web browser. Purpose: Display the flash clients and connect the flash client to Java mission server.
- Microsoft project. Purpose: Used for visualizing software design with GanttProject graph.
- Subversion SVN. Purpose: Used for project documentation version management.
- Microsoft Excel. Purpose: To calculate and summarize the working hours of group members.
- Moodle forum. Purpose: Used to discuss the project management issues.

The methods: when we tried to make the existing VRMOSS system running, we utilized five methods to try to solve the libraries compatibility problems:

- Visual Studio 2005 + modern libraries. Library headers used: Studierstube 3.3, Qt 4, Coin3D 2.5.0. Description: We encountered a lot of compatibility problems with the modern library headers. Studierstube 3.3 has a totally new concept for input device tracking (OpenTracker). After removing all references to tracking from the code, a lot more compatibility issues with Studierstube and Coin came up.
- Eclipse IDE+MinGW compiler. Library headers used: Studierstube 3.3, Qt 4. Description: Solarstube was originally created using Visual Studio 4.2, so there was no makefile to base an Eclipse project on. Some of the code is not standard C++ so it won't compile with Gnu compiler without modifications.
- Visual Studio 6 + partially ancient libraries Library headers used: most library headers from ASH CD (ancient) Description: We installed Visual Studio 6 and attempted to export the Solarstube Visual Studio project to an Eclipse-compatible makefile. Visual Studio exports to an nmake makefile which is not compatible with Gnu make. We found a script for translating nmake makefiles to Gnu makefiles but couldn't make it work. We didn't give too much attention to get Solarstube compiled with VS 6 since we soon found how to properly open Solarstube project files in VS 2005 and got further with that way.

- Studio 2005 + ancient libraries Library headers used:library headers from ASH CD (ancient) Description: Compilation was successful after some modifications to code using Math.h functions. Linking fails because of missing static lib-files (Qt 2.20, SoQt). We found Qt 2.30(Free-ware, not GPL), which should be backwards compatible with Qt 2.20 but static library file names were totally different from those defined in the project. For this approach to continue, we are waiting for a set of lib files from Delta. We have also tried to find a version of SoQt that we could compile against Qt 2.30 but unsuccessfully so far.
- Visual Studio 2005 + ancient libs & WinAPI Library headers used:library headers from ASH CD (ancient) Description: This is our latest attempt to build Solarstube. There are 209 errors generated during compilation.

14.4 Project phases and development model

We divided our project into three main phases: (1) rebuilding the old VRO-MOSS system, (2) adding haptic device functionalities and (3) add other functionalities. The sequence order of these phases should not be reversed, which means that the project phase one is the basic and precondition for other phases.

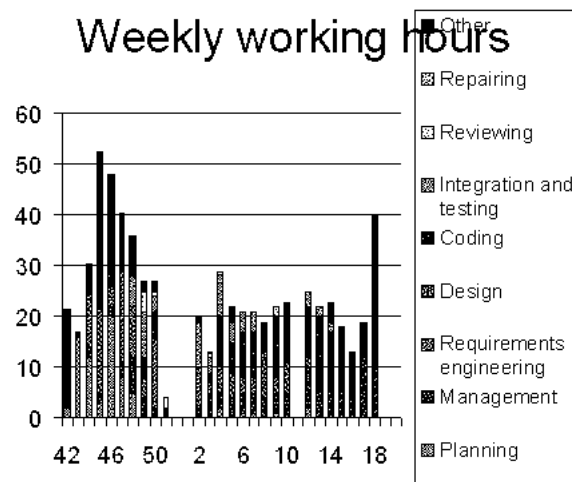


Figure 49: Weekly workload of the project.

In each phase, we apply 'Iterative approach' as the development model so that each project phase should go through inception, elaboration, construction and transition four step.

Unfortunately, the project phase one is the precondition task for subsequent tasks and we failed to implement this project phase which reason is mentioned in section one. The unfinished tasks, limited resources and schedule missing restricted us from doing the rest two project phases.

14.5 Conclusions

Although we did our foremost to fulfill our responsibility, the project failed at the end. We analyze and self-examine the project process and conclude several reasons which causes the failure.

- Resource limitation: we have five members at the beginning, but three of them quited at the beginning or in the middle. The remaining two members have a full time job which result in that they can not devote more time in this project each week.
- Project topic: the project topic is challenge one. It involves three huge tasks for our project team. We spent most time on understanding and rebuilding the old system, but the effort is trivial.
- Lack of suitable measure to handling unforeseen risks: During the system integrating, there were always unforeseen and unpredictable problems got us, although we spent much time on those problems, the effect is trivial. So I think missing a reasonable measure to this issue is important factor that make our plan schedule late.

14.6 Acknowledgement

Thanks a lot for our supervisor, Mr. Timo Poranen, Mr. Ilari Kajaste, our clients Roope Raisamo and Rami Saarinen. In addition, thanks for all the members in our group, although our project barged up against several obstacles inside the project, they never give up and do their foremost to solve the problems.

14.7 Statistics

The total number of working hours is more than 708 hours. The weekly working hours chart is shown in the figure 49. Because this project failed, therefore there maybe some statistics missing for the tables.

Table 72: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
2+3	Iterative	6.10.2007	16.5.2008	220	708	0.64

Table 73: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Design	Code	Integ. and test	Reviews	Re-pair	Study	Other	Total
Hours	185	116	7	120	59	9	0	-	158	654
%										
Usability	-	-	-	-	-	-	-	-	-	-
Total	193	116	7	120	59	17	0	-	196	708

Table 74: Design outcome.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
-	1	6	-	-	-

Table 75: Inspection findings.

	Preliminary analysis	Project plan
Pages and/or screens	10	37
Preparation time	31-10-2007	19-11-2007
Inspection time	1-11-2007	26-11-2007
Findings	7 mistakes	22 mistakes
Used time / findings	1 hour	2 hours

Table 76: Project's documents.

Document	Pages	versions
Preliminary analysis	10	0.1 - 1.0
Project plan	37	0.1 - 1.1
Project's usability plan	-	-
Requirements specification	-	-
Design plan	-	-
User interface document	-	-
Test plan	-	-
User's guide	-	-
Error report	7	0.1 - 1.0
Usability test report	-	-
UI Heuristic evaluation report	-	-
Final report	32	0.1 - 1.0
Final story	4	1.0
Weekly reports	26	
Inspection reports	4 (2+2)	
Total	120	

Table 77: Project's codelines.

Language	-
LOC	-
SLOC	-
Reused code	-
Reused and modified code	-
Classes	-
Functions	-
Code revisions	-

Table 78: Productivity metrics.

PM	LOC / PM	Models / PM	RS & DES pages / PM	LOC pages / PM	Total pages / PM
4.66	-	2.58 (12 Models)	-	-	25.75 (120 pages)

15 EDP2008

The purpose of our project is to create the web user interface for Mr Juhani Norris' dictionary of medical vocabulary in English (1375-1550). We are continuing the work of one of last year's project groups which completed converting the database from paradox DBMS to PostgreSQL DBMS and creating the printed version of the dictionary. Our task is to create the online version of the dictionary.



Figure 50: Our logo.

15.1 Overview

Our application is a set of PHP scripts that run on a web server with Apache. The information is fetched from the PostgreSQL database and processed in order to be shown to the user. The system will be built on top of the CakePHP framework (www.cakephp.org). The CakePHP framework is an open source rapid development framework for PHP that implements well known design patterns in a Model-View-Controller architecture. There are five main functions in the administrative area of our system: Users, Texts, References, Quotes and variants, and Auxiliary tables. The Users function manages the users of the administrative areas with some operations: create a new user, edit a user, delete a user, etc. With Texts, References, Quotes and variants functions, the user can add, view, edit and delete each item chose from the table of all items. The auxiliary tables function includes nine sub-functions which help the user to add, edit, view and delete auxiliary fields of database. Our main purpose is creating the administrative area for our client to add, edit and view items his database.

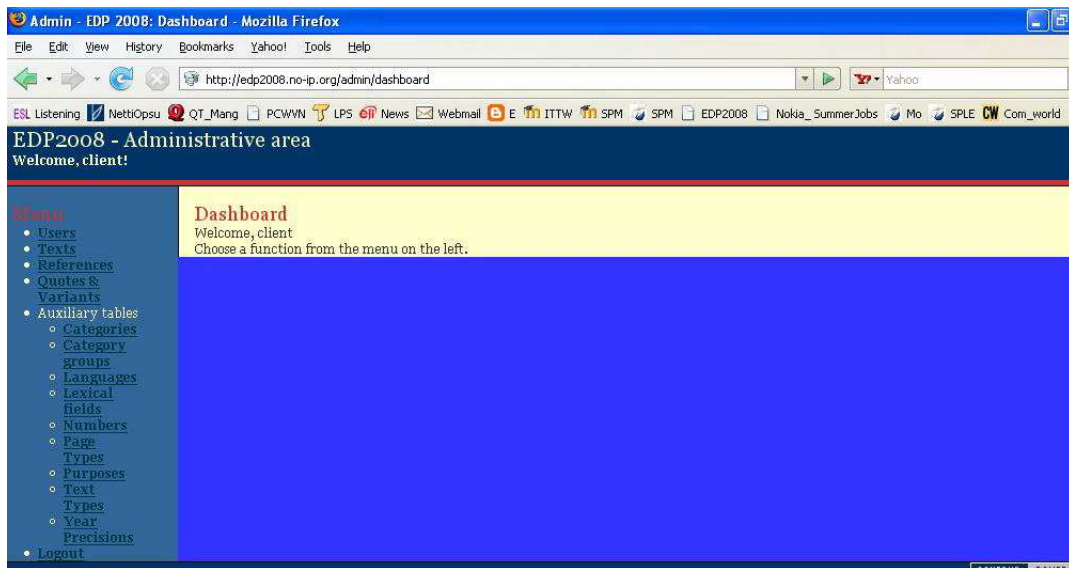


Figure 51: Main area in our system.

15.2 Organisation and management

Our group consisted of two project managers and three developers.

- Tung Doan: Project manager.
- Anil Kumar: Project manager.
- Maira Brandao Carvalho: PHP coder, Designer, Webpage admin.
- Ville Rahikainen: Tester, HTML coder.
- Matias Pyykönen: PHP coder, Code server admin.

One member quited the group in the beginning of the spring semester.

People connected to the project:

- Course lecturer - Timo Poranen
- Supervisor in Autumn semester - Ilari Kajaste
- Supervisor in Spring semester - Timo Poranen
- Steering group (consisting of the client, course lecturer and the project managers)
- Two project managers and three developers and

- Two members from the course usability group: Matti Huokko (Matti.Huokko@uta.fi) and Riki Kawakami (Riki.Kawakami@uta.fi).

One more member from usability group, Jenni Päckilä (jenni.pakkila@uta.fi), joined in our group from February 14, 2008. Matti participated in creating the User Interface specification for the first iteration. Matti, Riki, and Jenni created the User Interface specification for the second iteration.

The project group held weekly meetings throughout the whole project; from October 2007 to May 2008. Attendance was obligatory for the team members depending on the urgency and importance of the tasks at the time. In the last month (May 2008), when members are quite clear about their tasks, we arrange weekly meetings through online Gtalk conference which saves time for all members. The project manager wrote down the meeting minutes and decisions made during the meeting, and distributed them as a to-do list to the project team by uploading them on the group's webpage.

Weekly reports were distributed by the project manager to the project supervisor, client representatives and project team each Monday night during the development process of our system. There are 26 reports: 12 reports in the autumn semester and 14 reports in the spring semester.

During the project duration there have been four inspection meetings with the client and supervisor. These meetings' purpose was to review the following documents: Preliminary analysis, project plan, requirements specification for the whole system, all necessary documents for the first iteration (requirement specification, user interface specification, implementation plan, and testing plan). The inspection meeting for the documents of the second iteration is not held because they are agreed and understood by all developers and the client. The outcomes of these meetings were new versions of the above documents as well as feedback from client and supervisor for project proceedings.

The group did not face communication problems and the team members cooperated smoothly between them. However, we met communication problem with usability members when sometimes they did not answer emails from us and we really did not know how to contact with them.

15.3 Methods and tools

Electronic dictionary is a web application using database. Hence, model used to develop this application is MVC (Model - View - Controller), where the model is the representation of the information that the application operates, so it is the database of medical dictionary, the view is the user interface, so it is the actual HTML page, the controller is the code which

handles the input event from user interface and accesses the model (<http://en.wikipedia.org/wiki/Model-view-controller>). The programming language is PHP and the database management system is PostgreSQL. The list of tools that will be used for the server running the production environment will be following:

- Apache 2.2.4
- PHP 5.2.3
- CakePHP framework
- PostgreSQL 8.2.5
- Subversion 1.4.4
- PhpPgAdmin and/or pgAdmin

For development the tools below are going to be used:

- Eclipse 3.2.2 or equivalent
- PHPEclipse plugin
- Subclipse plugin

15.4 Project phases and development model

Our development model is an incremental development with two iterations. The whole development process includes preliminary analysis, project plan, requirement specification, (design1, implementation1, testing1), (design2, implementation2, testing2), product delivery. The reason we choose this model because our new system is really big and we should divide into smaller parts in order to reduce the complexity. In the first iteration, we implement more simple functions first and then when developers are familiar with their tasks in this context, we will implement complex functions. In addition, with this model, we can release some versions soon and the client can check how the system runs.

15.5 Conclusions

The software project management is really big and headachy course. Although you only get from 6 to 12 ECTS, you spend much time and effort in order to pass it. I got much experience from this course. First of all,

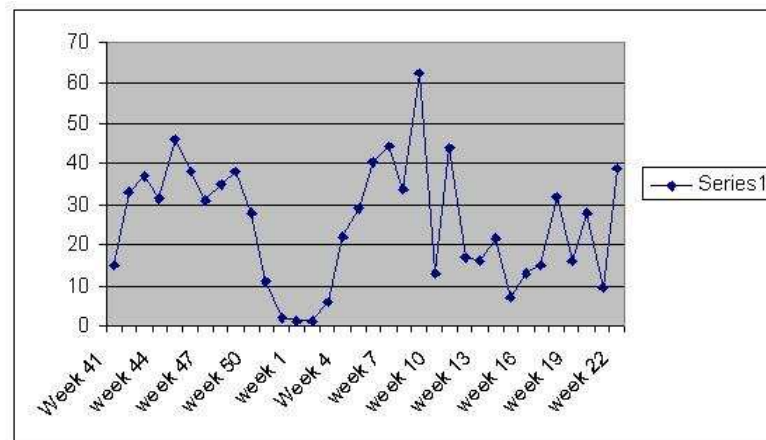


Figure 52: Weekly workload of the project.

which I studied did not come from techniques, programming skills. It mostly came from management skills and experience in real software projects. Skills in managing a team with members from different culture, experience and behavior are very useful for me in the near future. I understood that management is more complex than technology. Secondly, I learned how to develop a software system through all phases from gathering and creating requirement specifications from the client to implementing and testing the final system. It helps me have a wide view about the development of software systems and familiar with various tasks when working in practice. Lastly, it is not a course, it is very close to practice. After passing this course, I get more confident to participate in software projects when working with different roles: business analyst, software coder, software tester. In addition, I understood deeply and clearly principles and methods in software development which I studied from the course "theory of software project management". In summary, it is the most worth course for my programme: master programme in software development.

It was the first time I played a role as a project manager hence I lacked experience to make the project more success. Here are some things what to do better next time:

- Create strict rules at the beginning of the development process and encourage all members follow them.
- Make an efficient and clear schedule with detailed information about

the tasks based on your careful estimation about your resources: The skills of development, the time duration needed to finish them, etc.

- Members in your team should finish their tasks on time. It is very important because if your team delays some tasks, all other tasks will be delayed and it will take a lot of time for you to reschedule your project plan.
- Communication is also very important. Try to keep communication in your team smoothly and efficiently. It helps members understand each other and understand the whole things in developing your new system.

15.6 Statistics

Table 79: General project information.

Team size	Dev. mod.	Start date	End date	Days	Hours	Hours / (Days * Team size)
2+3+2	Incremental	15.10.2007	31.5.2008	195	852.5	0.87

Table 80: Group effort by activity.

Activity	Plan. and man.	Req. spec.	Des-ign	Code	Integ. and test	Rev-iews	Re-pair	Study	Other	Total
Hours	274.5	81.5	44	137.5	19	37.5	0	94	164.5	852.5
Percent	33	10	5	16	2	4	0	11	19	100

Table 81: Project's codelines.

Language	PHP
LOC	2134
Reused code	0
Files	50
Code revisions	46

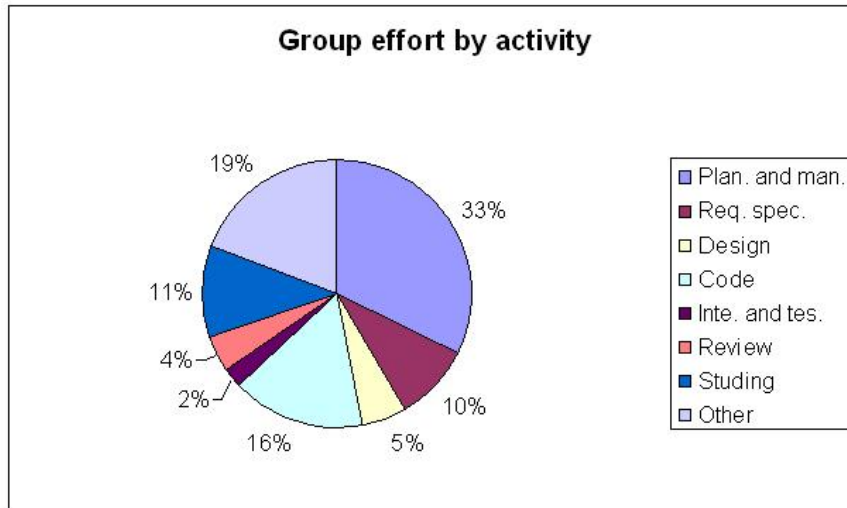


Figure 53: Group effort by activity.

Table 82: Project's documents.

Document	Pages	versions
Preliminary analysis	1	5
Project plan	4	27
Requirements specification	9	60
Design plan	4	14
User interface document	7	63
Test plan	6	40
Test report	1	4
Final report	2	20
Final story	1	8
Weekly reports	26	
Inspection reports	8	
Total	275	