Timo Poranen (ed.)

Software Projects 2009-2010

DEPARTMENT OF COMPUTER SCIENCES
UNIVERSITY OF TAMPERE

D-2010-14

TAMPERE 2010

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER SCIENCES
SERIES OF PUBLICATIONS D – NET PUBLICATIONS
D-2010-14, DECEMBER 2010

Timo Poranen (ed.)

Software Projects 2009-2010

DEPARTMENT OF COMPUTER SCIENCES
FIN-33014 UNIVERSITY OF TAMPERE

ISBN 978-951-44-8331-8
ISSN 1795-4274

Preface

This report contains project stories of 15 software development projects and one usability team. The students came from Project Work and Software Project Management courses. The stories describe what kind of experiences groups got during their project. In the end of each story there are project statistics.

Table 1 gives an overview of projects, project type (WWW, mobile or standalone application), client (CS department, non-commercial associations, company or Demola), development model, group size (number of managers + number of developers + number of usability team members) and working hours of the project. Projects which are marked with an asterisk (*) further developed an existing software.

Table 1: General project statistics.

Project	Type	Client	Dev. Mod.	Group	Hours
Uteam	-	CS Dept.	-	2+10	2428
ProSTTY	WWW	Assoc.	Scrum	2+4+1	1559
Laitteisto*	WWW	CS Dept.	Incremental	2+5+1	1068
Evaluointi	WWW	CS Dept.	Iterative	2+5+1	1085
Polku	WWW	Assoc.	Scrum	2+5+1	1097
Stud files	WWW	Assoc.	Scrum	2+5+1	1236
Officemapper	WWW	Company	Scrum	2+4+1	895
Disco*	Application	CS Dept.	Scrum	2+5+1	870
CMM	WWW+Mobile	Company	Scrum	2+5+1	1251
Mediabank	WWW+Mobile	Demola	Scrum	2+5+1	1268
Majava2*	WWW	CS Dept.	Iterative	2+5+1	995
Tambic	WWW	Univ.	Iterative	2+4+1	1383
Birthdata	Mobile	Demola	Scrum	2+5+1	1687
Context	WWW+Mobile	Demola	Iterative	3+6+1	1303
Cognitive	WWW+Mobile	Demola	Scrum	3+5+1	1040
Tracker	WWW	CS Dept.	Scrum	2+3+1	533

Although ten projects applied Scrum development model, they had one major difference when compared to standard Scrum: daily scrum meetings were mainly organized virtually using IRC or similar real-time messaging systems.

Stud files –project received Pitky (Information Processing Association of Tampere Region, http://www.pitky.fi) award. The award was given during Test days 2010 (see http://www.cs.tut.fi/tapahtumat/testaus10/).

Table 2 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 year 2005.

Table 2: Course statistics 2005-2010

Academic year	Projects	Usability teams	PW students	SPM students	Average project size
2005-6	19	1	98	8	1008
2006-7	18	2	87	34	1089
2007-8	14	1	70	29	997
2008-9	10	1	60	39	1643
2009-10	15	1	80	34	1151

During the course we started to use projectWiki for maintaining course and project related documentation: https://projectwiki.cs.uta.fi. The wiki also contains some articles on project management and project management tools, including lists of end-products currently in use, course related publications and course related videos:

- https://projectwiki.cs.uta.fi/wiki/Finished_projects
- https://projectwiki.cs.uta.fi/wiki/Course_publications
- https://projectwiki.cs.uta.fi/wiki/List_of_project_videos

Course staff thanks our clients and students for great projects.

Timo Poranen

Tampere, December 2010

Preface i
Usability team 1
ProSTTY 7
Laitteistotietokanta 12
Evaluointi 19
Polku 23
Stud Files – Best for the Bitches 30
OfficeMapper 35
DisCo 40
Content Manager Mobile 46
MediaBank 53
Majava2 59
TamBiC Project Story 63
BirthData 77
Rich context information sharing in Facebook 83
Collective cognitive state of people 91
Tracker 97

Usability team

Overview

The purpose of the Usability team (Uteam) is to have one group which focuses on the usability cases of different projects. The first usability team was organized in the project work course in years 2005-2006, and since then, it has evolved to suit the usability-related needs of different projects better.

The members of the group are experts in different areas of usability, and this year, the team consisted of two project managers and ten usability team members, who all study interactive technology as their major subject. Each of the usability team members worked in one or two projects, in which they helped in all usability related matters, for example organizing usability tests and designing and evaluating the user interface.

Different kinds of workshops were also a big part of the Usability team's work. We organized six workshops, in which we worked as a team, solving projects usability problems and doing heuristic evaluations. This kind of team work is the main idea of the project, one person simply cannot see all the problems in the user interface.

The communication between the projects and the Uteam was mainly done via the Uteam members. However, during the course, the Uteam project managers contacted other project's managers and asked about their experiences with the usability team. The feedback was mainly positive.

The Uteam's client, Timo Poranen, attended two document inspections, and he was informed about the states of the project in weekly reports.

The Uteam gave the students of Interactive technology an excellent opportunity to try out their learned craft in practice. This gave them an insight into what an usability experts workload include and how it may vary in various projects. A chance to observe all the projects was invaluable. Besides working as an educational tool, the work of the Uteam experts was greatly appreciated by the projects they worked on. Overall the Uteam project was once again a success and should be continued in following years.

Organisation and management

Project Managers:

- Ari Koivuniemi
- Jonna Paananen

Usability team members:

- Anu Leppälampi, worked on: MediaBank, Content manager mobile
- Auvo Salmenharju, worked on: Laitteistotietokanta
- Jaakko Mäntysaari, worked on: ProSTTY
- Jarkko Rinkinen, worked on: Tauchi, Disco
- Joonas Jokiniemi, worked on: Cognitive state of people, Polku
- Leena Kylliäinen, worked on: Officemapper
- Minna Heinonen, worked on: Majava2, Stud Files

- Sanna Kangas, worked on: Tracker
- Santeri Saarinen, worked on: Birthdata
- Juhani Vainio, worked on: Context, TamBic

Figure 1: Usability team members from left to right: Anu Leppälampi, Juhani Vainio, Leena Kylliäinen, Auvo Salmenharju, Sanna Kangas, Jonna Paananen, Minna Heinonen, Jarkko Rinkinen, Joonas Jokiniemi, Ari Koivuniemi, Jaakko Mäntysaari and Santeri Saarinen.

Methods and tools

The methods used by the usability team are listed below.

- Heuristic evaluation
  - Heuristic evaluations were done in the Uteam workshops.
  - They were found to be very useful, and lots of problems were found and corrected because of them.
- Usability testing
  - This year, six projects organized usability tests in the usability lab.
  - The results of the tests were used to improve the product.
- Workshops
  - The workshops are extremely useful; the group support really seems to count.
  - Working together in workshops also builds the team spirit.
- Peer reviews
  - There is always something one doesn't notice having stared at her work for too long. This is when a fellow expert's views come in very helpful.
- Usability analysis
  - The very first usability related document for the project. Things keep changing as project progresses but doing this properly gives the assigned expert a proper insight into project's scale.
- Brainstorming
  - Good ideas always help. Sometimes you need less perfect ones first to get into the good stuff.
- Mock-ups

    - Mock-ups helped team to get a better view on the project.
- Use cases
    - Use cases were written in some projects.
    - They were found to be very useful in planning the usability tests. On the other hand, one of those projects didn't use them at all.
- Expert analysis
- User studies
    - User studies were extremely helpful in the projects, where the idea was still growing during the implementation. A Must in every agile software development project.
- User interface plans
    - Basically the documentation where all the usability expert's work come together.
    - Keeps the track of UI iterations and why certain actions work they do.

Project phases and development model

The project didn't have any special development model, due to the nature of the project. The team's first meeting was 19.09.2009, and the project members were assigned to their projects during the following weekend. First assignment for the team was to write a preliminary analysis, which was inspected on 24.09.2009. The team members started their projects by analyzing their projects usability needs and writing a usability analysis-document. These documents were supposed to be finished by the deadline of the project plan, which was being written at the same time and inspected on 30.10.2009.

The workshops started right away, the first one was held on 01.10.2009. In most of the workshops, we arranged heuristic analysis on 1-3 projects. These meetings took about 2-3 hours per project, and they were found to be very useful in finding usability problems. We also had two workshops that were a bit different. All team members had 30 minutes to present their project to others, tell what they have done in the project, what are they going to do next, and are they going to do usability tests on their project. These workshops were a great way to get to know other projects and find out what others have done.

The team members helped their projects on designing the user interface, creating prototypes, writing UI documents, and testing the usability of the product. Six projects decided to arrange a usability test in the usability lab, and they were mainly done on January and February. Also a user study was done on one project. Some projects were behind the schedule, and because of that, their plans about testing the product's usability had to be changed. This year we didn't have any problems in reserving the usability lab, because the course started a bit earlier than last year.

Most of the final weeks of the course were spent on writing usability test reports and other documents relevant to the product. The important milestones of the project are presented in Table 3.

In overall, the project was very successful. We didn't have many problems, and most of the risks we tried to predict on the project plan, didn't occur. An unforeseen risk that did occur was that because some projects were behind the schedule, some members didn't have enough work to do. This was fixed by letting those members to help other projects on arranging the usability tests.

Date	Milestone
19.09.2009	First Meeting
24.09.2009	Inspection of the preliminary analysis
01.10.2009	First workshop
30.10.2009	Inspection of the project plan
25.11.2009	Second workshop
08.12.2009	Christmas party
09.12.2009	Presentation day
20.01.2010	Presentation of the usability lab
21.01.2010	Third workshop
09.02.2010	Fourth workshop
16.02.2010	Fifth workshop
26.02.2010	Final presentation
01.03.2010	Sixth workshop
12.03.2010	Final meeting

Table 3: Important milestones.

Experiences

This year's Project work course started already in September which, from the Usability team managers' point of view, meant extra work, since we needed to make sure our team members were allocated to client projects and got into them early enough to fit in. The starting phase of the grouping has proven to be essential in making everybody feel they fit in. Usability team managers received info on our group members before the other projects' managers which made this allocation possible and the most usability experts were able to participate in their client project's first meeting. This practice should be maintained.

Because of the changed schedule this year there was no overlapping with the using of the usability lab. There were few close calls, but they can be avoided if the usability testing of the projects is planned well ahead. With this timetable, the managers should encourage the team members to start planning for the testing around December. All the testing should be finished by halfway through February.

There could be more direct communication between the Uteam managers and the other projects' managers. The information flows just fine via Uteam members, but there may be cases of misunderstanding or scheduling events like heuristic evaluation. One or two emails during the course may be sufficient to remind them we care.

This year there were fifteen projects which all required a usability expert. This lead to a large Uteam also, total of ten experts and two managers. This is starting to be the upper limit of the group size - it's getting difficult to find big enough meeting rooms for starters.

This year's amount of experts could handle maybe few more projects, but that would result in growing work amount. If the work amount of given project could be estimated more precisely by the client, it would help in determining if an expert should be assigned for a sole project or can she handle two.

Statistics

In this section of this document, the Usability team's work is presented in statistics. In Table 2, start- and end dates of the projects and the overall amount of hours are presented. In Figure 2, the group effort by activity and the total number of hours per week, are presented. Finally, the produced documents are presented in Table 3.

Team size	Dev. model	Start date	End data	Days	Hours
2+10	N/A	19.09.2009	12.3.2009	175	2428,5

Table 4: General project information.

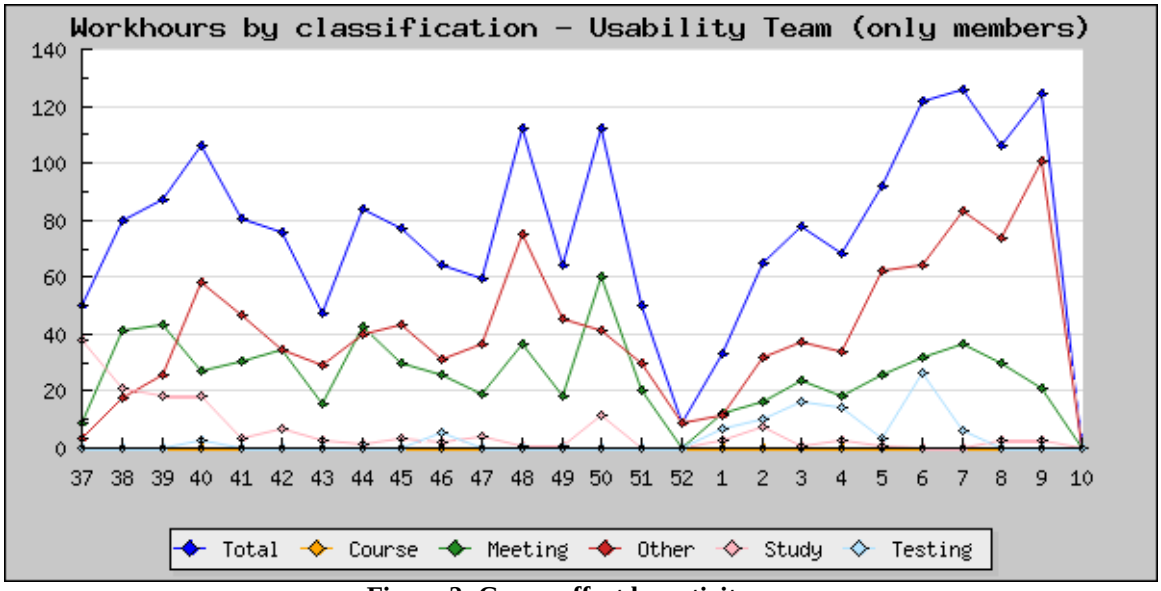


Figure 2: Group effort by activity

Document	Pages	Versions
Preliminary analysis	38	7
Project Plan	61	11
Usability analysis	143	20
Design plan	84	7
User interface document	81	12
Usability test plan	51	13
Usability test report	118	13
Final report	44	2
Project's story	14	3

Weekly reports	21	1
Heuristic evaluation	36	6
Survey report	8	5
Total	**699**	**100**

Table 5: Documents.

ProSTTY

Yleistä

Tuote on Sosiaali- ja terveydenhuollon tietojenkäsittely-yhdistys STTY:lle tuotetut uudistetut www-sivut, jonka käyttöliittymän avulla yhdistyksen jäsenet sekä yhdistyksen hallituksen jäsenet voivat hyödyntää STTY:n palveluita. Erilaisille käyttäjäryhmille on luotu ryhmien tarpeiden mukaiset palvelut. Jäsenten ja hallituksen palveluihin on pääsy käyttäjätunnistuksen avulla. Projektissa tuotettiin jäsenrekisteri, johon projektin toteutuksen aikaan kuului yli 200 jäsentä. STTY:n käyttöliittymään tehtiin jäsenrekisterin muutosten ja jäsenmaksuseurannan työkaluja, joiden käyttämiseen ei tarvita ohjelmointitaitoja.

Kuva 3. Sivuston etusivu.

Organisointi ja hallinta

Projektilla oli ohjausryhmä, joka seurasi projektin etenemistä ja osallistui tarpeen mukaan katselmointeihin. Projektin ohjausryhmään kuuluivat kurssin vetäjä Timo Poranen Tampereen yliopiston Tietojenkäsittelytieteiden laitokselta, projektipäälliköt, STTY:n edustajat sekä käytettävyysryhmän projektipäällikkö Jonna Paananen. Projektiryhmässä oli 6 projektityöntekijää: Sami Kiviharju, Simo Pönni, Ristomatti Salomaa, Heikki Säily ja käytettävyysryhmän edustajana Jaakko Mäntysaari sekä 2 projektipäällikköä: Sari Kurimo ja Tiina Taivalantti. Projektipäälliköt olivat vastuussa asiakasyhteydenpidosta ja ohjausryhmän tilaisuuksien sekä projektin viikkopalaverien järjestämisestä. Projektiryhmästä muodostui projektin työskentelyn myötä luontevasti kaksi "tiimiä", pääasiassa tekninen toteutusryhmä ja käytettävyys/ulkoasutiimi.

Kuva 4. projektiryhmän jäsenet (kuvasta puuttuu Ristomatti).

Menetelmät ja käytetyt ohjelmistot

Ohjelmoinnissa oli käytössä PHP-, XHTML- ja CSS. Tietokantana toimi MySQL. Sivuston sisällönhallintajärjestelmäksi valittiin Joomla!. Projektin aikana tuotettujen materiaalien versionhallintaan käytimme Subversionia (SVN). Ryhmän jäsenten käyttämät Subversion-asiakasohjelmat olivat vapaasti heidän valittavissaan, mutta käytimme lähinnä TortoiseSVN- ja RapidSVN-ohjelmia.

Yhteisten dokumenttien työstämiseen käytimme Google Docsia. Yhteensopivuusongelmien välttämiseksi sovimme kaikkien ryhmän jäsenten käyttävän OpenOfficen versiota 3.1. Julkaistut dokumentit tallennettiin pdf-muotoon.

PHP-, XHTML- ja CSS -dokumenttien tekemiseen kukin käytti valitsemaansa tekstieditoria tai WWW-/ohjelmistokehitysympäristöä. Käytössä olivat ainakin Notepadin 5.1-versio, Notepad++ ja Emacs-sovelluksen 23.1.1-versiota. WWW-sivuston graafiset elementit luotiin Adobe Photoshopilla, GNU Image Manipulation Programilla (GIMP) ja Inkscapella.

Ohjelman toimintaa kehityksen aikana testattiin XAMPPlite-ohjelmistopaketin 1.7.1-versiolla. Testaukseen käytettiin ainakin Mozilla Firefoxin ja Internet Explorerin viimeisimpiä versioita, Google Chromea, Operaa ja Safaria. Käytössä oli myös Debiania pyörittävä virtuaalipalvelin jolle asennettiin PHP:n, Apachen ja MySQL:n uusimmat versiot.

Työryhmän julkisten ja sisäisten asioiden raportoimiseen, ajankohtaisista asioista tiedottamiseen ja ryhmän jäsenten tuntikirjanpitoon sekä testauksessa havaittujen virheiden kirjaamiseen käytimme Wikiä.

Ryhmän aikataulujen yhteensovittamisessa käytimme Doodlea ja Google Calendaria. Päivittäisessä yhteydenpidossa oli IRC-kanava #prostty@IRCNet. Ryhmän sisäiset tärkeät tiedotteet lähetettiin myös sähköpostitse. Asiakkaalle luotiin Wikiin oma osio määrittelyiden täydentämiseen ja tarvittavien lisätietojen antamiseen.

Projektin kehitysmalli ja vaiheistus

Projektin kehitysmalli oli Scrum. Projekti jaettiin kolmen viikon pituisiin sprintteihin, joita oli 7 kpl. Mallia muokattiin opiskelijaympäristöön sopivammaksi, korvasimme päivittäiset tapaamiset viikoittaisella tapaamisella, jossa käytiin läpi edellisen viikon aikana tehdyt tehtävät ja suunniteltiin seuraavan viikon tehtäviä. Kehitysmalli soveltui meille hyvin projektin aiheen huomioon ottaen. Sprinttien aikana saatiin katselmointeihin aina jotain uutta näytettävää. Aina ei saatu aikaan uutta toimivaa ominaisuutta katselmointiin mennessä, mutta joka tapauksessa työ eteni jokaisen sprintin aikana. Sprinttien vaihtuessa järjestettiin katselmointitilaisuudet ja käynnistettiin seuraava sprintti.

Projektikurssilaisten työskentelytavaksi muotoutui projektin edetessä workshop-tyyppiset tapaamiset viikkopalaverien lisäksi. IRC-kanava toimi myös koko projektin aikana hyvänä kommunikointivälineenä.

ProSTTY AIKATAULU 0.5 Päivitetty 3.2.2010

Viikkonro	40	41	42	43	44	45	46	47	48	49	50	51	52	53	1	2	3	4	5	6	7	8	9
SPRINTIT																							
Sprintti 1	1. SPRINTTI			Syysl																			
Sprintti 2				2. SPRINTTI																			
Sprintti 3				Syysl			3. SPRINTTI																
Sprintti 4				Syysl						4. SPRINTTI													
Sprintti 5				Syysl								5.			SPRINTTI								
Sprintti 6				Syysl													6. SPRINTTI						
Sprintti 7				Syysl																7. SPRINTTI			

Kuva 5. Projektin aikataulu

Katselmoinnin aihe	Katselmointiajankohta	Timo Poranen osallistui alla mainittuihin tilaisuuksiin:
Esitutkimus: Versio 0.1	21.9.2009	Esitutkimuksen katselmointi
Projektisuunnitelma versio 0.1	9.10.2009	Katselmointitilaisuus I
1. sprintin katselmointi	16.10.2009	
2. sprintin katselmointi	6.11.2009	Katselmointilaisuus II
3. sprintin katselmointi	27.11.2009	
4. sprintin katselmointi	8.12.2009.	Katselmointitilaisuus III
5. sprintin katselmointi	15.1.2010	
6. sprintin katselmointi	5.2.2010	Katselmointitilaisuus IV
7. sprintin katselmointi	23.2.2010	
Loppukeskustelu	25.2.2010	Loppukeskustelutilaisuus

Kuva 6. Katselmointiaikataulu.

Johtopäätökset

Projektityökurssilla tuli kaikille osallistujille paljon uusia haasteita vastaan. Projektipäälliköt tutustuivat ensimmäistä kertaa projektin johtamisen kiemuroihin ja projektityöntekijät saivat kosketusta oikean asiakkaan kanssa toimimiseen ja oikean tuotteen toteuttamiseen. Projektissa käytetty Joomla! –sisällönhallintajärjestelmä oli kaikille uusi asia ja sen opetteluun menikin projektissa melko paljon aikaa. Töiden jakaminen oli aluksi hieman ongelmallista, mutta sujui lopulta melko luontevasti, kun projektiryhmän jäsenet oppivat tuntemaan toisensa ja taitonsa paremmin. Kaiken kaikkiaan projektityökurssi oli kaikille opettavainen kokemus. Ajoittaisesta motivaation laskusta ja väsymyksestä huolimatta projektista jäi ennen kaikkea positiivinen mielikuva.

Tilastot

Tämä kappale sisältää projektin tärkeimpiä tilastoja taulukkomuodossa ja kuvina. Useat dokumentit työstettiin Google Docsissa, jonka vuoksi niistä ei ole erikseen versionumeroita.

Ryhmän koko	Kehitysmalli	Alkupäivä	Loppupäivä	Päiviä	Tunteja
2+4+1	Scrum	14.9.2010	26.2.2010	166	1559

Taulukko 6. Yleistä tietoa projektista.

Tehtävä	Tunnit	%
Projektin suunnittelu ja johtaminen	269	17 %
Vaatimusmäärittely	12,5	1 %
Suunnittelu	263,0	17 %
Ohjelmointi	220	14 %
Integrointi ja testaus	201	13 %
Katselmoinnit	81,5	5 %
Korjaus	84,5	5 %
Opiskelu	180,5	12 %
Muu	247	16 %
Yhteensä	**1559,0**	**100 %**

Kuva 7. Työtunnit tehtävittäin.

Vaatimuksia	Käyttötapauksia	Käyttöliittymän näyttökuvat	Tietokanta-kaaviot	Tietokantataulut
54	2	116	7	16

Taulukko 7. Vaatimuksia ja korkean tason suunnitelmia.

Dokumentti	Sivuja	Versioita
Esitutkimus	10	2
Projektisuunnitelma	29	7
Kevyt käytettävyyskartoitus	16	1
Käytettävyystestisuunnitelma	1	1
Testaussuunnitelma	12	1
Testiraportti	3	1
Käytettävyysraportti	26	2
Käyttöönottosuunnitelma	2	1
Loppuraportti	39	1
Projektikertomus	6	1
Viikkoraportit	22	-

Taulukko 8. Dokumentit.

Kieli	PHP, XHTML, CSS, JavaScript
LOC	9980
SLOC	9284
Luokat	12
Funktiot	287

Taulukko 9. Arvioitu koodirivien määrä.

Laitteistotietokanta

Yleiskatsaus

Projektin tuloksena toteutettu tuote on Tampereen yliopiston tietojenkäsittelytieteiden laitoksen käyttäjätunnusten, laitteiden, ohjelmistolisenssien ja lainattujen asennusmedioiden päivitetty hallintajärjestelmä. Tarkoituksena oli korjata aikaisemmin toteutetusta järjestelmästä vikoja ja lisätä siihen toivottuja ominaisuuksia. Käyttäjinä on tietojenkäsittelytieteiden laitoksen atk-tukiryhmän lisäksi laitoksen henkilökunta, sekä ne laitoksen hallinnon ja sidosryhmien edustajat, joiden tulee saada tai kyetä muuttamaan järjestelmässä käsiteltäviä tietoja.

Kuva 8. Sisäänkirjautuminen.

Organisaatio ja projektinhallinta

Projektin asiakas oli Tampereen yliopiston tietojenkäsittelytieteidenlaitos.

Kurssin edustaja oli:

- Timo Poranen

Asiakkaan edustajana projektissa oli:

- Timo Taipalus

Projektipäälliköinä olivat:

- Jussi-Pekka Koro
- Teppo Lindell

Projektiryhmänjäset olivat:

- Toni Järvilehto
- Henri Kynsilehto

Kuva 9. Projektiryhmän jäsenet (vasemmalta lukien: Henri, Heikki, Auvo, Teppo, Ville, Toni ja Jussi. Kuvasta puuttuu Pauli).

- Pauli Lammi
- Heikki Metsäpuro
- Ville Vierimaa

Käytettävyysryhmän jäsen oli:

- Auvo Salmenharju

Menetelmät ja työkalut

Toteutuksessa käytettiin PHP-, HTML-, Javascript-, ja CSS-tekniikoita. Tietokantana käytetään PostgreSQL:ää. Koska projektin tarkoituksena oli jatkokehittää aiemman vuoden projektityökurssilla toteutettua järjestelmää, oli osa suunnitteluratkaisuista tehty puolestamme. Yksi näistä ratkaisuista oli, että käytössä ei ollut mitään ohjelmistokehystä vaan aiempi ryhmä oli toteuttanut kaiken omilla toteutusmalleillaan.

Koodien versionhallintaan käytettiin Subverionsia(SVN) ja suurin osa ryhmästä käytti tässä työkaluna TortoiseSVN-ohjelmaa.

Ryhmän eniten käytetty työkalu kurssin aikana oli dokuwiki-asennus, jonka avulla seurattiin ryhmän työtunteja ja koottiin dokumentaatio yhteen paikkaan. Dokumentaation tuottamiseen dokuwiki sopi erittäin hyvin, koska sen avulla oli helppo koota yhteen useammalle ihmiselle kirjoitettavaksi annettu dokumentti. Tämä helpotti myös projektipäälliköiden elämää antamalla yhden työkalun lisää projektin edistymisen seuraamiseksi. Dokuwikiin on tarjolla myös monia hyödyllisä laajennoksia, kuten laajennos jolla voi tuottaa wikisivusta .odt-tyyppisen tiedoston ja laajennos jolla on mahdollista tehdä wikistä myös offline-tilassa selailtava. Nämä kaksi laajennosta ovat hyödyttäneet projektikertomusta kirjoitettaessa ja projekti-cd:tä koottaessa.

Yhteydenpitoon käytettiin sähköpostilistaa ja tämä havaittiin tehokkaaksi keinoksi koko ryhmän yhteiseen yhteydenpitoon. Käytössä oli myös IRC-kanava missä oli helppo ratkoa ongelmia nopeasti vähän enemmän aikaa vievän sähköpostienvaihdon sijaan.

Projektin vaiheet ja kehitysmalli

Projektissa käytettiin inkrementaalista kehitysmallia. Tämä oli ainakin meidän projektiryhmällemme sopivin kehitysmalli, koska olisi ollut mahdotonta saada ryhmän jäsenet esimerkiksi päivittäisiin Scrum-tapaamisiin. Projektille varattu aika oli myös niin lyhyt, että esimerkiksi vesiputousmalli hylättiin heti alkuunsa.

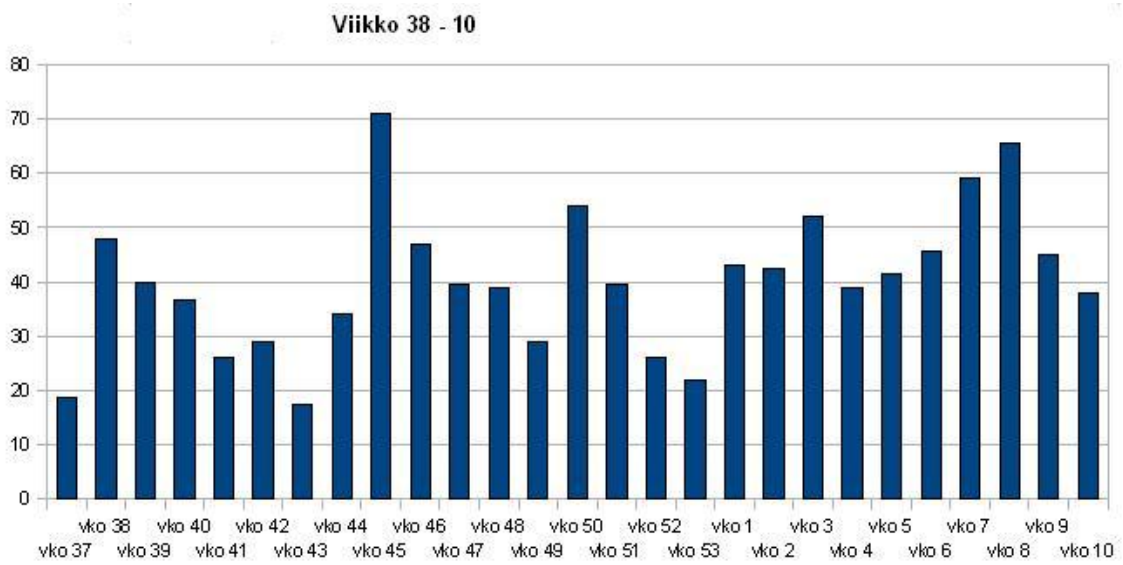
Katselmoinnin aihe	Katselmointiajankohta
Esitutkimus	25.09.09
Projektisuunnitelma	08.10.09
1. Katselmointi	12.11.09
2. Katselmointi	15.01.10
3. Katselmointi	18.02.10
Projektikertomus	12.03.10
Loppuraportti	09.03.10
Projekti-CD	12.03.10
Lopputapaaminen	09.03.10

Taulukko 10. Projektin aikataulu.

Ryhmä kohtasi ainoastaan ennakoituja riskejä ja ennakoimattomia riskejä ryhmä ei kohdannut yhtäkään. Ryhmään liittyvistä riskeistä toteutui ainoastaan pieni ajankäyttöongelma mistä ei syntynyt varsinaista ongelmaa vaan tilanne hoidettiin töiden jakoa ja aikataulutusta hieman muuttamalla. Tuotteen riskeistä toteutui pohjalla olleen järjestelmän aiheuttamat ongelmat, mutta se ei vaikuttanut lopulliseen tuotteeseen millään näkyvällä tavalla. Viimeinen ennakoitu riskinä olivat aina silloin tällöin ilmaantuneet ongelmat laitoksen palvelinten kanssa. Nämä ongelmat olivat kuitenkin enimmillään päivän mittaisia.

Inkrementti	sisältö
1 (7.9-27.9.2009)	Esitutkimus, aloituspalaveri, Projektisuunnitelman sisällön kirjoitus. Versionhallinnan ja testiympäristön pystytys.
2 (28.9-11.10.2009)	Projektisuunnitelman kirjoittaminen ja katselmointi. Katselmointi viimeistään 9.10 (palautus 5 pvä. ennen) Käytössä olevaan järjestelmään perehtymistä
3 (12.10-1.11.2009)	Vaatimusmäärittelyn kirjoittaminen ja katselmointi. katselmointi viimeistään 30.10. Vikojen priorisointi ja vikojen korjaamista ja testaamista
4 (2.11-15.11.2009)	Henkilökohtaisten raporttien kirjoitus ja palautus moodleen 7.11. Vikojen korjausta.
5 (16.11-29.11.2009)	Havaittujen vikojen korjausta.
6 (30.11-13.12.2009)	Vikojen korjausta. Uusien ominaisuuksien toteutusta
7 (11.1-24.1.2010)	Vikojen korjaamista. Henkilökohtaisen raportin palautus (7.1.2010). uusien ominaisuuksien toteutusta.
8 (25.1-.7.2.2010)	Vikojen korjausta ja uusien ominaisuuksien tekeminen.
9 (8.2-21.2.2010)	Uusien ominaisuuksien tekemistä. Toteutus valmiina 15.2.
10 (22.2-6.3.2010)	Koodien viimeistelyä, loppuraportin kirjoittaminen ja loppuesitykseen valmistautuminen. Loppuesitys ja projekti CD:n luovutus asiakkaalle ja kurssinvetäjälle. Viimeisen henkilökohtaisen raportin palautus ennen lopetuspalaveria.

Kuva 10. Projektin inkrementit.



Kuva 11. Työtunnit.

Kokemuksia

Yleisesti ottaen omat kokemukseni tästä projektista ovat positiiviset. Näin pääasiallisesti siksi, koska projekti eteni kokonaisuudessaan suunnitellusti ilman suurempia takaiskuja. Projektin alussa projektiryhmän keskuudessa tuntui olevan valloillaan ajatus siitä onko projektiin suunniteltu sisältö riittävän laaja. Huoli töiden riittävyydestä kuitenkin osoittautui melko nopeasti turhaksi, koska projektiryhmän

jäsenet löysivät ahkerasti uusia parannuskohteita sekä uudet ominaisuudet myös tarkentuivat ajan myötä ja työllistivät riittävästi.

Projektiin valittu inkrementaalinen malli soveltui hyvin tämän tyyppiseen projektiin, jossa projektiryhmän jäsenet eivät ole päivittäin tekemisissä toistensa kanssa. Inkrementaalinen kehitys eroaa iteratiivisesta kehityksestä periaatteeessa siten, että inkrementaalisessa kehityksessä lopputuote tehdään pienissä palasissa (osatoimitukset) ottamatta kuitenkaan kantaa kuinka inkrementti itsessään toteutetaan (esim. minivesiputous). Iteratiivisessa kehityksessä taas keskeinen ajatus on toistaa samaa asiaa uudelleen ja uudelleen aina rautalankamallista tuotantovaiheen ohjelmistoon. Projektin alussa tehty oletus siitä, että ketterämmät mallit kuten Scrum tai XP tai vastaava eivät sovellu tämän tyyppiseen projektiin, jossa ei voida järjestää päivittäisiä tapaamisia kasvokkain osoittautui oikeaksi.

Dokumentaation tuottamisessa ei ilmennyt ongelmia ja mikä parasta, se ei jäänyt yksittäisen henkilön vastuulle vaan jokainen teki oman osansa. Projektin esitykset sujuivat esimerkillisesti. Katselmoinnit saatiin pidettyä ajallaan ja sujuivat hyvin. Kaikki suunnitellut ominaisuuden saadaan toteutettua ajallaan ja projekti-CD saadaan koottua ennen eräpäivää, joten voidaan hyvällä syyllä todeta, että projektiryhmä on onnistunut tavoitteessaan.

Tilastot

Ryhmän koko	Kehitysmalli	Aloitus pvm.	Lopetus pvm.	Päivät	Tunnit
2+5+1	Inkrementaalinen	09.09.09	12.03.10	-	1067,5

Taulukko 11: Projektin tiedot.

Aktiviteetti	Suunnittelu ja hallinnointi	Vaatimusmäärittely	De-sign	Toteutus	Integrointi ja testaus	Katselmoinnit	Korjaus	Opiskelu	Muut	Yhteensä
Tunnit	327	5,5	37,5	237	11,5	15,5	62	213	158,5	1067,5
%	30,7	0,5	3,5	22,2	1,1	1,5	5,8	20,0	14,8	100
Käytettävyys	-	-	-	-	-	-	-	-	-	133,5
Yhteensä	327	5,5	37,5	237	11,5	22	62	213	158,5	**1201**

Taulukko 12: Ryhmän tunnit.

Vaatimusten määrä	Sivut	Käyttötapaukset	Näyttöjä	Tietokantaaviot	Tietokantataulut
42	43	-	25	1	11

Taulukko 13: Vaatimukset.

Sivut	Kaaviot	Luokkakaaviot	Sekvenssi-kaaviot	Tilakaaviot	Muut kaaviot
23	2	-	-	-	-

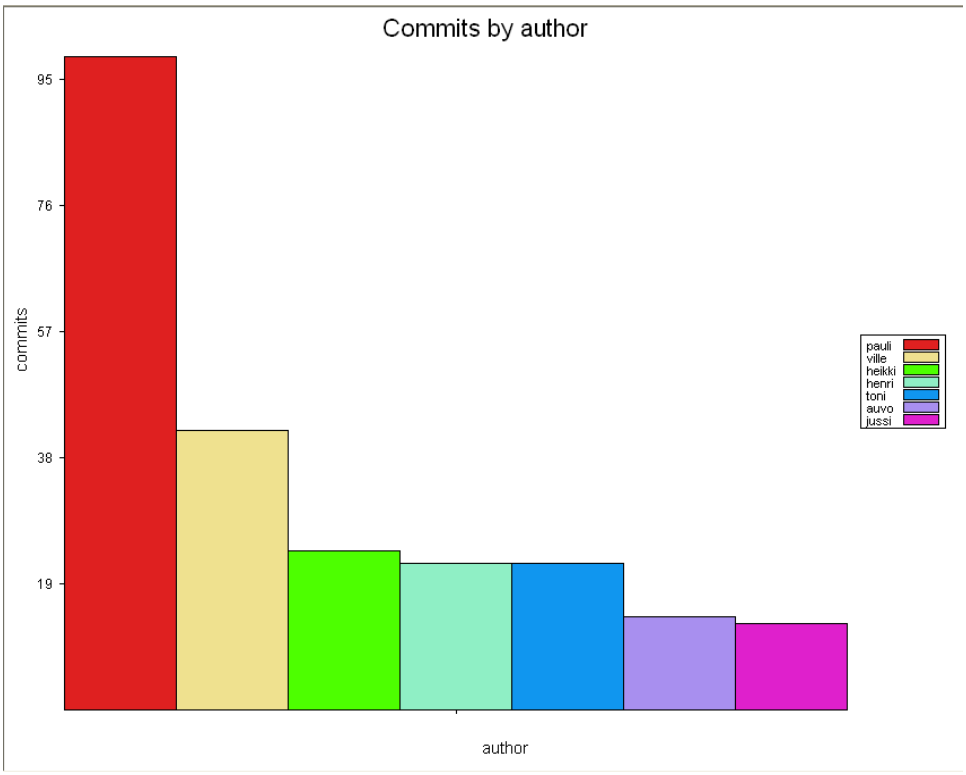
Taulukko 14: Suunnittelu.

Dokumentti	Sivut	Versiot
Esitutkimus	5	14
Projektisuunnitelma	10	133
Käytettävyysanalyysi	-	-
Vaatimusmäärittely	47	128
Toteutussuunnitelma	-	-
Käyttöliittymäsuunnitelma	-	-
Testaussuunnitelma	6	50
Testiraportti	-	-
Heuristinenarviointi	8	1
Loppuraportti	9	95
Projektikertomus	9	10
Viikkoraportit	1	24

Taulukko 15: Dokumentaatio.

Kielet	PHP, JavaScript
LOC	7265
SLOC	4815
Uudelleenkäytetty koodi	-
Muokattu koodi	-
Luokat	17
Funktiot	144
Koodiversiot	235

Taulukko 16: Koodi.



Kuva 12. Versionhallinnan käyttö.

Evaluointi

Overview

Goal of the project was to design and develop a tool, Survey Creator, to create surveys for research. The product also gathers data from test participants and processes the data to reusable format. The client was Markku Turunen on behalf of TAUCHI unit.

The main requirements for Survey Creator were:

* Easy drag-and-drop tool to create surveys and survey pages.

* API to create custom UI components for survey.

* Support for people with impaired vision.

* Possibility to reuse old survey form templates on future researches.

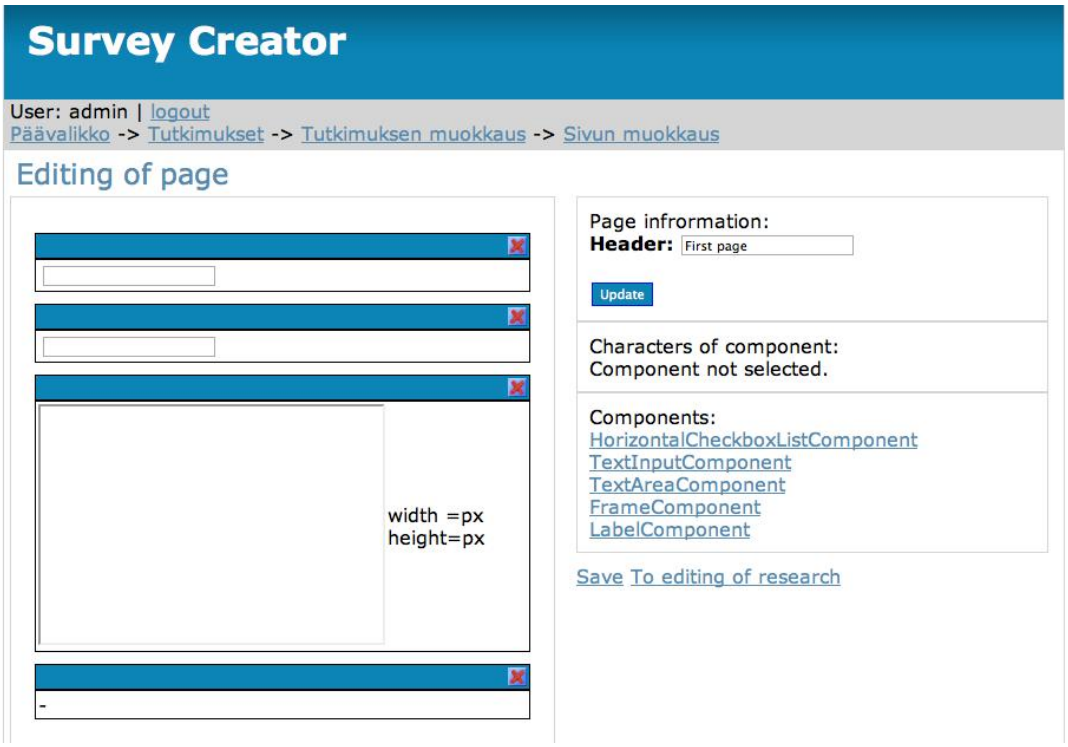


Figure 13. Survey Creator's page editing screen.

Organisation and management

Project management:

Arttu Ekholm, arttu.ekholm@uta.fi

Mikko Arminen, mikko.arminen@uta.fi

Project members:

Tuomas Granlund, tuomas.granlund@uta.fi

Tommi Ritola, tommi.ritola@uta.fi

Samuli Häyrynen, samuli.hayrynen@uta.fi

Antti Torkko, antti.torkko@uta.fi

Usability specialist:

Jarkko Rinkinen, jarkko.rinkinen@uta.fi

Figure 14. *The project group (from left: Arttu Ekholm, Samuli Häyrynen, Tuomas Granlund, Jarkko Rinkinen, Mikko Arminen, Tommi Ritola, Antti Torkko).*

Methods and tools

The project was developed with PHP. Some functionality was programmed with JavaScript and JQuery. PostgreSQL was used for the database, although the product is also MySQL compatible. Version control was done with Subversion.

Project group held weekly meetings to keep track of the progress. Design and programming workshops were also held when needed.

Communication occurred mostly via email and IRC. Occasionally, telephone was also used. Documents were stored in DokuWiki wiki.

emacs	Some members found this editor useful.
Firebug	Really good tool for debugging web applications
Google Docs	Very easy way to maintain working hours
DokuWiki exporting	Simple to install and use, but lacked WYSIWYG and PDF features.
NetBeans	Useful all-around editor and IDE
IRC	Useful for communicating between group members
JQuery	Good for creating drag-n-drop functionality
Subversion	Version control. Simple to use.

Project phases and development model

We chose iterative waterfall as our development model. The project was divided to four phases:

On the first phase, architecture for the software was designed and developed.

Second phase was about designing and developing the survey creation tool. At the beginning of this phase, simple UI prototype was developed. The prototype was expanded to contain complete functionality, and rest of the project was built on the prototype. Second phase ended on December 2009

Third phase began on January 2010. It contained design and development of view for participants.

On fourth phase we developed functionality for data export. Final testing, bug fixing and usability evaluation were also done on this phase. This phase ended on April 2010.

The project suffered a major drawback when one of members left the project during the second phase.

Experiences

In retrospect, the project group agreed on that programming should be started on very early phase. It is also important to produce working prototypes early and often to keep client and group members motivated.

Statistics

Team size	Dev. model	Start date	End date	Days	Hours
2+4+1	iterative	1.9.2009	14.4.2010	225	1085

Table 17: General project information.

Acti-vity	Planning and management	Req. specification.	De-sign	Code	Integ-ration and testing	Re-views	Repair	Study	Other	Total
Hours	231.5	39	66.5	448	5	30	12	122.5	132	1085
%	21.3	3.6	6.1	41.3	0.5	2.8	1.1	11.3	12.2	100.0
Usabi-lity	-	-	-	-	-	-	-	-	-	-
Total	**231.5**	**39**	**66.5**	**448**	**5**	**30**	**12**	**122.5**	**132**	**1085**

Table 18: Group effort by activity.

Number of requirements	Pages	Use-cases	UI screens	Database diagrams	Database tables
23	22	9	2	1	1

Table 19: Requirements and high-level design outcomes.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
122	-	-	-	-	-

Table 20: Design outcomes.

Document	Pages	Versions
Preliminary analysis	5	1
Project Plan	8	1
Usability analysis	11	1
Requirements specification	22	1
Design plan	13	1
User interface document	4	1
Test plan	7	1
Test report	10	1
Usability test report	2	1
Final report	7	1
Project's story	8	1
Weekly reports	-	-

Table 21: Documents.

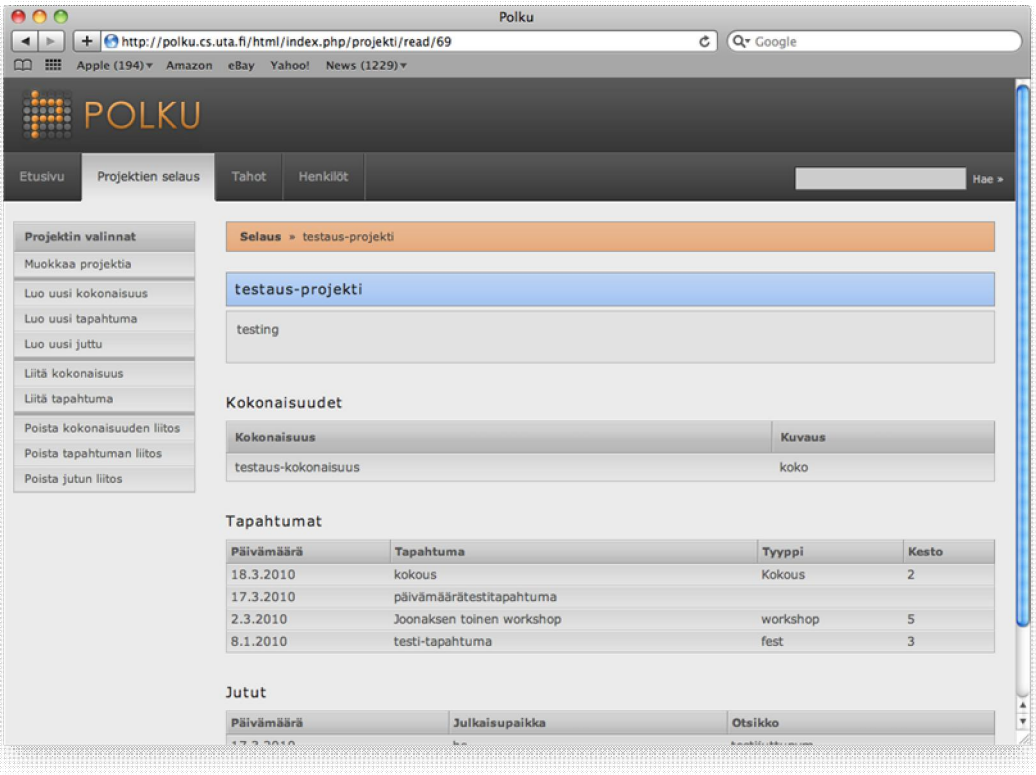
Language	PHP, XHTML, CSS, JavaScript
LOC	8033
SLOC	-
Reused code	-
Reused and modified	-
Classes	63
Functions	-
Code revisions	236

Table 22: Codelines.

Polku

Johdanto

Projektin tarkoituksena oli luoda web-sovellus pitkäaikaisten yhteistyöprosessien ja -projektien seurantaan Helsingin kaupunginosayhdistykset ry:lle. Sovelluksella piti olla mahdollista tallentaa, muokata ja poistaa tietoa erilaisista tapahtumista, niihin liittyvistä tahoista ja henkilöistä sekä näiden välisistä suhteista. Edellä mainittuihin tuli myös pystyä liittämään juttuja (esim. mainos tai artikkeli jossain lehdessä) ja huomioita. Tapahtumia tuli myös pystyä ryhmittelemään kokonaisuuksiksi. Sovelluksessa tuli olla ristiinlinkityksiä eri objektien välillä, mikä tekikin siitä varsin haasteellisen. Tuotteen tuli mahdollistaa tapahtumien ja toimijoiden listaamisen ja auttaa näin ollen projekteihin liittyvässä raportoinnissa.



Kuva 15. Sovelluksen käyttöliittymä.

Projektin organisointi ja johtaminen

Henkilöstö ja roolit:

Projektipäälliköt

Janne Pihlajaniemi

- Vastuualueet: viikkopalaverien vetäminen ja tilanvaraus niitä varten ym. ryhmän tapaaminen Tampereella

Antti Jämsén

- Vastuualueet: tiedottaminen asiakkaalle, asiakastapaamiset Helsingissä, viikkoraportit

Työryhmä

Maria Hartikainen

- Dokumentit, sovelluksen testaus

Nina Tyni

- Tietokanta, sovelluksen koodaus

Jorma Laajamäki

- Tietokanta, sovellusalustan ylläpito

Panu Tunttunen

- Käyttöliittymä, sovelluksen koodaus

Pekka Kallioniemi

- Käyttöliittymä, sovelluksen koodaus

Käytettävyysryhmän edustaja

Joonas Jokiniemi

- Käyttöliittymä, käytettävyys

Kuva 16. Polku-projektin henkilöstö.

Projektin johtaminen

Projektipäälliköt sopivat yhdessä johtamisen linjauksista, mutta Janne Pihlajaniemi johti itse ryhmää Tampereella, koska toinen projektipäällikkö asuu Helsingissä ja pääsi vain harvoin käymään Tampereella. Ryhmäläisiltä kysyttiin projektin alussa halua tehdä eri projektin osa-alueita ja roolit jaettiinkin aika lailla näiden toivomusten mukaisesti. Eri osa-alueille löytyikin vapaa-ehtoisesti halukkuutta ryhmäläisiltä, joten tehtävien jako sujui hyvin helposti.

Johtaminen oli lähinnä tukevaa ja ryhmäläisille annettiin melko suuriakin vapauksia toteutuksen suhteen, kunhan vain toiminnallisuuksien vaatimukset täyttyivät. Myös asiakkaan edustajaa konsultoitiin hyvin usein, jotta mahdolliset virheet saatiin mahdollisimman nopeasti korjattua. Asiakas osallistuikin hyvin aktiivisesti projektiin, mikä helpotti ryhmän työskentelyä.

Välineet ja menetelmät

Projektissa hyödynnettiin GoogleWikiä, joka on monipuolinen Wiki-työkalu projekti-työskentelyyn. Kotisivu löytyy osoitteesta: http://code.google.com/p/polku-projekti/. Projektin kotisivulta löytyy myös projektin tuottamat dokumentit ja sitä käytettiin mm. tiedostojen ja tiedon jakamiseen ryhmäläisten kesken. Tämä osoittautuikin hyvin hyödylliseksi projektin edetessä.

Tuntilistaukset ylläpidettiin Google-dokumenttien taulukkolaskennassa, johon jokainen ryhmäläinen kävi joka viikko viimeistään sunnuntaina kirjaamassa edellisviikon tuntinsa. Google Groupsissa on luotu sähköpostilista, jonka avulla tietoa saadaan nopeasti lähetettyä ryhmän kesken. Keskitetty dokumentti, jota kaikki pystyivät muokkaamaan oli hyvä ratkaisu. Välillä vain tahtoi unohtua tuo tuntien merkkaaminen.

Ohjelmointiin kukin ryhmäläinen sai käyttää haluamaansa ohjelmistoa, joista suosituin taisi olla Eclipse.

Sovellukseen käytettiin myös CodeIgniter-nimistä PHP Frameworkia, joka tarjosi paljon valmista koodi-alustaa sovellukselle ja helpotti huomattavasti ohjelmointityötä. Sen käyttö todettiin varsin tarpeelliseksi ja hyödylliseksi.

Testaamiseen löydettiin myös kätevä ohjelma Selenium, joka on selaimeen (Mozilla Firefox) asennettava Plug-In. Kyseiseen ohjelmaan pystyy nauhoittamaan esim. lomakkeen täyttämistä ja ajamaan sitten kyseistä nauhoitusta läpi yhä uudelleen. Selenium nopeutti hyvin paljon testaamista, kun joka kerralla ei tarvinnut tehdä samoja toimintoja uudestaan.

Yhteydenpito projektin aikana tapahtui pääasiallisesti sähköpostin kautta ja tapaamisissa, mutta projektilla on myös oma IRC-kanava, jota hyödynnettiin etenkin ohjelmointivaiheessa.

Sovellus toteuttiin LAMP-ympäristöön, joka oli asiakkaan toivomusten mukainen.

- Palvelin: Linux Ubuntu Server 8.04.3 LTS
- Web-palvelin: Apache
- Tietokanta: MySQL 5.0.51a-3ubuntu5.4
- Ohjelmointikieli: PHP 5.2.4-2ubuntu5.7, Suhosin-Patch

Kehitysympäristönä projektilla toimi tietojenkäsittelytieteiden laitoksen tarjoama virtuaalipalvelin, jossa oli palvelinohjelmistona perusasennettu Debian GNU/Linux 5.0. Palvelin toimi hyvin kehitysalustana eikä sen käytössä ilmennyt ongelmia.

Projektin vaiheistus ja kehitysmalli

Kehitysmalli

Projektimme kehitysmallina käytettiin Scrumia. Käytännön syistä johtuen Scrumia ei kuitenkaan noudatettu sillä tarkkuudella kuin yritysmaailmassa, koska päivittäisiin tapaamisiin ei ollut mahdollisuutta.

Vaiheistus

Sprintti	*Sisältö*	*Viikot*	*Pääajankohdat*
#1: Projektin aloitus	– Projektin käynnistäminen – Aiheeseen tutustuminen – Ideointi – Projektisuunnitelma – Käytettävyyskartoitus	38-41	Katselmointi: 07.10.09 Esitutkimus: 22.09.09 Projektisuunnitelma: 04.10.09 Käytettävyyskartoitus: 09.10.09
#2: Vaatimusmäärittely ja suunnittelu	– Ohjelmointityökalujen asentaminen – Ohjelmointityökaluihin tutustuminen – Käyttöliittymän suunnittelu – Tietokannan suunnittelu – Tietokannan ER-malli – Käyttöliittymä-suunnitelma – Vaatimusmäärittely	42-45	Katselmointi: 06.11.09 Vaatimusmäärittely: 03.11.09
#3: Proton muodostus ja ohjelmoinnin aloitus	– Tietokannan muodostaminen – Käyttöliittymän muodostaminen – Staattisen proton tekemisen aloitus – Ohjelmoinnin aloitus	46-48	Katselmointi: 27.11.09
#4: Proton muodostus ja ohjelmointia	– Staattinen proto valmis – Testidata syötetty tietokantaan – 8% valmiina	49-51	Katselmointi: 16.12.09 Projektiesitys: 09.12.09
#5: Ohjelmointia,	– Ohjelmointi jatkuu	1-3	Katselmointi: 22.01.10

katselmointia ja korjausta	– Katselmointia ja korjaamista asiakkaan kommenttien perusteella – 17% valmiina		Testaussuunnitelma: 21.01.10
#6: Ohjelmointia ja katselmointia	– Ohjelmointi jatkuu – Session hallinnan toteuttaminen – Testaus-suunnitelman tekeminen ja katselmointi – 77% valmiina	4-6	Katselmointi: 15.02.10
#7: Ohjelmointia ja testaamista	– Ohjelmointi jatkuu vielä, lähinnä kuitenkin hienosäätöä eikä uusien ominaisuuksien toteuttamista – Bugien korjaus – Testaamista – Sovelluksen toimittanen asiakkaalle heidän sisäistä testausta varten (koska mm. hieman erilainen ympäristö kuin kehitysympäristö) – Loppujen dokumenttien kirjoittaminen – 93% valmiina	7-9	Katselmointi: 04.03.10 Testausraportti 1: 02.03.10 Testausraportti 2: 03.03.10 Projektikertomus: 07.03.10 Loppuraportti: 07.03.10
Projektin päättäminen	– Lopputapaaminen asiakkaan kanssa – Projekti-CD – 100% valmiina	11	Lopputapaaminen 18.03.10

Riskit

Ennalta määritellyiltä riskeiltä vältyttiin onneksi, joista pahimpia olisi ollut projektihenkilöiden kurssin keskeyttäminen. Yhtenä riskinä pidettiin riittämätöntä ohjelmointikokemusta PHP-kielestä, jolla sovellus pääosin toteutettiin, mutta ryhmäläisten ahkeran opettelun vuoksi myös ohjelmointi sujui hienosti. Oikeastaan mitään yllättäviä riskejä ei matkan aikana tullut vastaan.

Kokemukset projektista

Projekti sujui loppujen lopuksi varsin hyvin. Projektiryhmäläiset olivat aidosti kiinnostuneita projektin toteuttamisesta eikä motivaatiosta ollut puutetta. Projektin onnistumisen kannalta oli tärkeää, että ryhmä tuli hyvin toimeen koko projektin ajan.

Erään haasteen projektille toi toisen projektipäällikön asuminen Helsingissä ja työskentely kokopäiväisesti. Projektin johtaminen kaipaa kuitenkin selkeästi myös paikallaoloa ja kasvokkain tapahtuvaa kommunikointia ryhmän kesken. Onneksi toinen projektipäällikkö pystyi vetämään viikoittaiset tiimipalaverit läpi, koska muutoin projekti ei todennäköisesti olisi onnistunut.

Meillä oli projektissa myös erittäin aktiivinen asiakas, mikä helpotti huomattavasti etenkin iteratiivisesti tapahtuvaa kehitystyötä. Me saimme nopeasti vastauksia ja kommentteja toteutettuihin asioihin jolloin niitä päästiin myös muuttamaan nopeasti halutunlaiseksi. Tietenkin lähtökohtana oli mahdollisimman tarkka vaatimusmäärittely jo ennen toteutuksen aloittamista, mutta käytännössä vaatimukset täsmentyvät kuitenkin vasta toteutuksen aikana. Meillä tällainen interaktiivinen työskentelytapa toimi hyvin.

Tilastot

Team size	Dev. model	Start date	End data	Days	Hours
2+5+1	Scrum	16.09.09	18.03.10	184	1097

Taulukko 23: General project information.

Acti-vity	Planning and management	Req. specification.	De-sign	Code	Integ-ration and testing	Re-views	Repair	Study	Other	Total
Hours	364	25	71	249	49	43	83	123	90	**1097**
%	33,18	2,28	6,47	22,7	4,47	3,92	7,57	11,21	8,2	100

Taulukko 24: Group effort by activity.

Number of requirements	Pages	Use-cases	UI screens	Database diagrams	Database tables
63	34	0	5	1	27

Taulukko 25: Requirements and high-level design outcomes.

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

Taulukko 26: Design outcomes.

Document	Pages	Versions
Preliminary analysis	14	v0.1, v1.0
Project Plan	32	v0.1, v1.0
Usability analysis	11	v0.1, v0.2, v0.3
Requirements specification	34	v0.1, v1.0, v1.1, v1.2
Design plan (3 documents)	32	v0.1, v0.2, v2.5
User interface document	-	-
Test plan	30	v0.1, v1.0
Test reports 1 and 2	1+1	v1.0 (both)
Usability test report	8	v1.0
Final report	19	v1.1
Project's story		
Weekly reports	23	v1.0

Taulukko 27: Documents.

Language	PHP, HTML
LOC	4734 (PHP) 3630 (HTML)
SLOC	8203
Reused code	0
Reused and modified	0
Classes	
Functions	
Code revisions	2180

Taulukko 28: Codelines.

Stud Files – Best for the Bitches

Overview

The general idea behind this program was to help all the dog breeders job in finding good and healthy match for they bitches. Earlier the selection was made mostly by intuition or based on outdated information that was available only in paper folders and was accessible to only a few breeders who managed to get hold of those folders.

Stud Files -program is available to everyone via Internet. It is easy to use and has correct data all the time. This will help the dog breeders job by providing wide search options and the ability to make test-matches between two dogs. All this leads to better breeding decisions and hence healthier dogs in the future when unfortunate matches can be prevented.

Figure 17. Example of test-match between a male and a bitch.

Organisation and management

There are eight members total in Stud files -team. The two project managers are Jaana Partanen and Jaakko Helenius.

The project group consist of developers Johanna Aittoniemi, Janne Leinonen and

Jussi Ampuja, the graphical designer Tomi Nukarinen and the database manager Henrik Mustonen. The usability team member for the Stud Files -team is Minna Heinonen.

Figure 18. The Stud Files -team. Members from left to right: Henrik Mustonen, Jussi Ampuja, Jaakko Helenius, Johanna Aittoniemi, Tomi Nukarinen, Jaana Partanen and Minna Heinonen (missing from the picture are Janne Leinonen and our customer liaison Jonna Paananen).

Methods and tools

Project documentation is held in Google Docs. This has been found very useful for many reasons. Most important of those reasons are that the documentation is reachable to everyone who had access to the Internet and since the documents are both created and modified in Google Docs there is no problems with different text-editing-programs or program versions. Project also has wiki pages but they are not really used because holding everything in one place in Google Dogs is more convenient.

Project communicates via email, irc and face-to-face -meetings. Even thought there is no match to a meeting with all the project members in present, the irc is also very effective communication channel because it allows almost real-time response and allows many mutual participants.

Project phases and development model

The Stud Files project is implemented using modified Scrum as its development model. The major difference to pure Scrum is that instead of daily meetings there are

weekly meeting. The project is divided into a start up sprint plus seven actual sprints.

The Start up sprint lasted 3 weeks, it started at 2.9.2009 and ended at 24.9.2009. The two major things in that sprint were the project introduction presentation at 16.9.2009 and the pre-study document inspection that was held at 21.9.2009 with lecturer Timo Poranen. In the inspection the problem area for the project was found sufficient and starting of the project got green light.

Sprint one lasted 5 weeks, it started at 25.9.2009 and ended at 30.10.2009. During that time the project plan document was created and it was inspected at 8.10.2009 with Timo Poranen. In the inspection the project plan was found adequate and no major changes to it were required. Also the program development started in sprint one.

Sprint two lasted 3 weeks, it started at 31.10.2009 and ended at 20.11.2009. The requirements specification document was created in that sprint even thought it was updated throughout the project. The light usability assessment document was created and the first personal report was returned in this sprint too.

Sprint three lasted 3 weeks, it started at 21.11.2009 and ended at 11.12.2009. The project presentation was held at 9.12.2009. The project review was held at 11.12.2009 with Timo Poranen. In the review it was determined that the project was well on schedule and the development work was progressing as was expected.

Sprint four lasted 4 weeks, it started at 12.12.2009 and ended at 8.1.2010. There was the Christmas holidays in this sprint so there was only very little happening. The second personal report was returned at 1.1.2010.

Sprint five lasted 3 weeks, it started at 9.1.2010 and ended at 25.1.2010. Most of this sprint was developing the program so that it would be ready for testing.

Sprint six lasted 3 weeks, it started at 26.1.2010 and ended at 15.2.2010. In this sprint the program development was completed and both general and usability testing was started.

Sprint seven will last 3 weeks, it started at 16.2.2010 and will end at 5.3.2010. There is the final project presentation at 26.2.2010. Some things that will still be done in this sprint are the final completion of the development, returning the third personal report, writing final report, burning project CD and holding the final meeting.

All in all the project passed off without any major problems. None of the estimated risk occurred during the project and nobody left the course in the middle. The only thing that caused worry was the fact that our customers liaison, Jonna Paananen, left the customer company, Arunica Oy, during December 2009. Problems were nevertheless avoided when Jonna announced that she could still continue as a customer liaison till the end of the project.

Statistics

Team size	Dev. model	Start date	End date	Days	Hours
2+5+1	Scrum	02.09.09	08.03.10	185	1236

Table 29: General project information.

Activity	Planning and management	Req. specification.	De-sign	Code	Integration and testing	Re-views	Repair	Study	Other	Total
Hours	290	36,5	127	239,5	63	28	70,5	122,5	171	1148
%	23	3	10	19	5	2	6	10	14	100
Usabi-lity	24				9			1,5	69,5	104
Total	314	36,5	127	239,5	70	26	60,5	124	238,5	**1252**

Table 30: Group effort by activity.

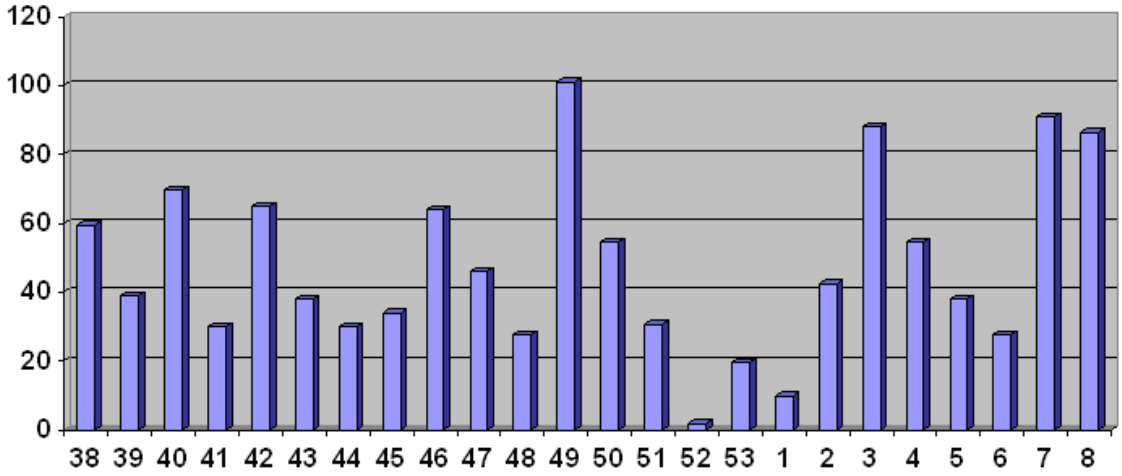


Table 31: Weekly hours.

Team member	Total hours
Jaana Partanen	187
Jaakko Helenius	172,5
Janne Leinonen	184
Jussi Ampuja	137
Johanna Aittoniemi	161,5
Tomi Nukarinen	113
Henrik Mustonen	193

Table 32: Total hours used by each group member.

UI screens	Database tables
11	13

Table 33: High-level design outcomes.

Document	Pages
Preliminary analysis	5
Project Plan	17
Usability analysis	6
Design plan	14
User interface document	21 (in 7 different documents)
Test plan	12
Test report	1
Usability test report	1
Final report	10
Project's story	6
Weekly reports	22

Table 34: Documents.

Language	PHP, HTML, JavaScript, CSS
LOC	4068
SLOC	3573
Functions	36

Table 35: Codelines (missing info from one developer from the SF-team).

OfficeMapper

Overview

OfficeMapper is an interactive office map and contact information tool made with Microsoft Silverlight and ordered by Sininen Meteoriitti Oy from Helsinki.

OfficeMapper provides an easy way to find locations and information about office workers. Basic information includes worker name, job title, team, phone number, email, birth date and a short description text.

The most special information is workers location in the office. Many offices are very large and there may be hundreds of workers. In this case it may be really difficult for a new employee to learn to remember and recognize co-workers names and faces. OfficeMapper adds a solution to this specific problem.

Organisation and management

Team members were:

- Timo Laak, Project Manager.
- Jesse Laanti, Project Manager.
- Heino Lindström, Software Architect.
- Maija Andersson, Software Designer.
- Johanna Vainio, Software Designer.
- Leena Kylliäinen, UI Specialist, from the UI Team.
- Petri Ikävalko, UI Designer.
- One team member quit the course.

Fig. 19: The Project Group. (Row 1: Timo & Jesse. Row 2: Johanna, Heino, Leena, Maija, Petri)

During the project we arranged regular face-to-face meetings with the group. In addition to this, we held weekly check-up meetings over IRC. This way we were able to keep track of task progress, as well as individual member statuses – i.e. how they felt about their work load, if they had other pressing matters preventing them from dedicating much time to the project and so forth. Of course in urgent matters we were also able to communicate by email and even telephone. All in all, our group did not need much prodding rigid direction as all members were very work-oriented.

Methods and tools

Silverlight is a Flash-like RIA (Rich Internet Application) platform and can be developed with programming languages that are supported in .NET, most commonly C#, which we also chose as a programming language. Silverlight UI is written in XAML (eXtensible Application Markup Language), which is XML-based language.

The most important development tools were Microsoft Visual Studio 2008 Professional and Silverlight 3 Software Development Kit. All program code and some of UI code was written in Visual Studio. Database was first run in Microsoft SQL Server 2008 Professional Edition on one of the team member's development server but due to hardware failure switched later into Microsoft SQL Azure cloud database service.

Version control was handled by Subversion acquired from Beanstalkapp.com. Database queries were written in LINQ (Language Integrated Query), which is a .NET framework component and makes writing of pure SQL unnecessary.

User interface and the visual layout was designed in Adobe Photoshop and written in Visual Studio and Microsoft Expression Blend 3.

Project phases and development model

Our project was loosely based on Scrum project management model. Development timeline was divided in seven sprints that were updated during the project when necessary.

Daily Scrum meetings were not organized because the development team was practically working remotely and independently during the project. Some of the team members were also working on their permanent job during the project.

Weekly development and review meetings were held on Thursdays. Each meeting consisted of monitoring the project state and work already done and tasks to be done before the next meeting. Meetings were also important working time to design program functionality, features, hunt for bugs and to learn from each other.

Occasional extra meetings were held if necessary. Progress was also monitored by weekly online meetings on the project's IRC-channel.

Project task list was never static. Tasks were updated if necessary to reflect changes in the project. However, main goals and the most critical features were static from the beginning of the project.

The most important foreseen risk was the project team's inexperience in software projects and required techniques. Therefore, it was seen crucial to start the project

with main focus on learning the techniques and planning tasks that could be easily done.

Other foreseen risk was team members' willingness to participate in project and take responsibility in completing given tasks. Luckily, only one team member decided to quit the project, so damage was minimal.

Unforeseen risks were software and hardware related. First, acquiring required software was problematical. We could not easily get any software from the university despite the fact that University of Tampere is a member in Microsoft's MSDN Academic Alliance developer network. Without other services for students, like IEEE Computer.org and Dreamspark.com we would not have been able to get required software and complete this project.

Second, our database server hardware crashed and left our team without a working database. Development was delayed until working database was restored, this time using Microsoft SQL Azure cloud database server, which caused some more problems. Visual Studio database tools could not be used with Azure and administering required the Microsoft SQL Server 2008 Professional R2 November Edition in minimum so database tools had to be upgraded.

We would recommend avoiding using own development server hardware or software and selecting a paid or freeware service instead. Always back up your data!

Experiences

Managing an inexperienced student project team with a new technique is not easy. Therefore it is really important to give them some time to learn and not to demand too much on early stage of the project. Focus is on learning, not expecting the team to complete the project with minimum working hours and resources like in the real world business.

Our experiences were mainly positive. The project was interesting and co-operation with the client was easy. The basic techniques were relatively easy to learn and support was available from the client if needed. The team progressed unbelievably well during the project and learned a lot of new things that cannot be learned on school bench.

Next time it would be important to give more focus on the team and the tasks. Make sure everyone has work to do. Monitor the working hours weekly. If someone is not working, ask for a reason. Everyone is expected to participate in the project.

Be aware if someone has problems and give support if needed. Be always aware how the project is progressing. What tasks are complete, what is still to be done? Is the project on schedule? Be ready to give a demonstration of the application to the client or the supervisor of the course. Track changes and bugs and make sure that bugs are fixed before the end of the project.

Use the best and most useful available tools to communicate with the team and to manage the project. Communication is the most important thing in team projects. Make sure the team members communicate with other members too, not just with project managers.

There are lots of documents to write. Find out which are important in your development model and start writing them early. Don't write long novels. Write only

what is essential. Briefs are important if decisions are made in meetings with the client.

Statistics

Team size	Dev. model	Start date	End data	Days	Hours
2+4+1	Scrum	September 14th 2009	March 14th 2010		895,25

Table 36: General project information.

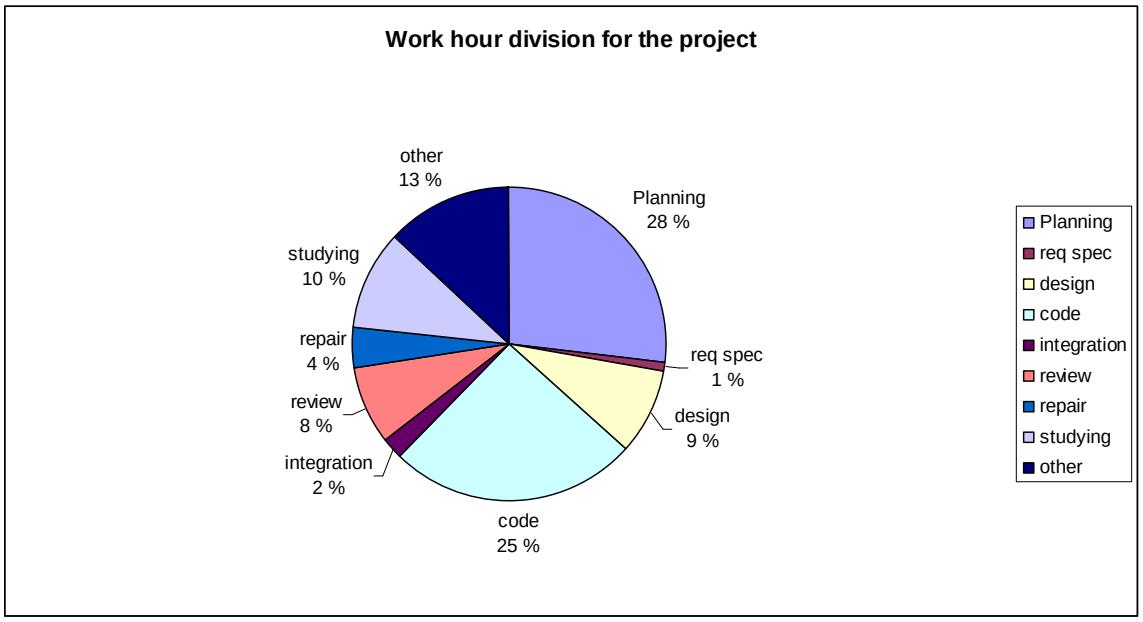


Table 37: Group effort by activity.

Number of requirements	Pages	Use-cases	UI screens	Database diagrams	Database tables
~20	N/A	-	7	1	8

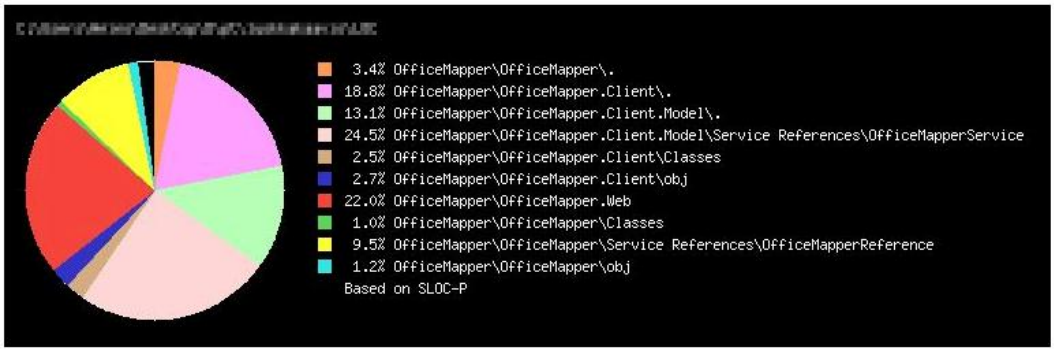
Table 38: Requirements and high-level design outcomes.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
N/A	0	3	0	0	2

Table 39: Design outcomes.

Document	Pages	Versions
Preliminary analysis	Done by Google Docs, no page count available	2
Project Plan	9	5
Usability analysis	9	4
Requirements specification	Done as a project backlog, no page count available	4
Design plan	-	-
User interface document	-	-
Test plan (both technical and usability)	2	1
Test report	-	-
Usability test report	7	2
Final report	24	4
Project's story	7	2
Weekly reports	~20 (approx one for each week of the course)	1 / report
Installation instructions	1	2

Table 40: Documents.



Overall		
Symbol	**Count**	**Definition**
Source Files	46	Source Files
Directories	532	Directories
LOC	11319	Lines of Code
BLOC	1905	Blank Lines of Code
SLOC-P	8998	Physical Executable Lines of Code
SLOC-L	5688	Logical Executable Lines of Code
MVG	878	McCabe VG Complexity
C&SLOC	4	Code and Comment Lines of Code
CLOC	416	Comment Only Lines of Code
CWORD	2652	Commentary Words
HCLOC	0	Header Comment Lines of Code
HCWORD	3	Header Commentary Words

Table 41: Codelines.

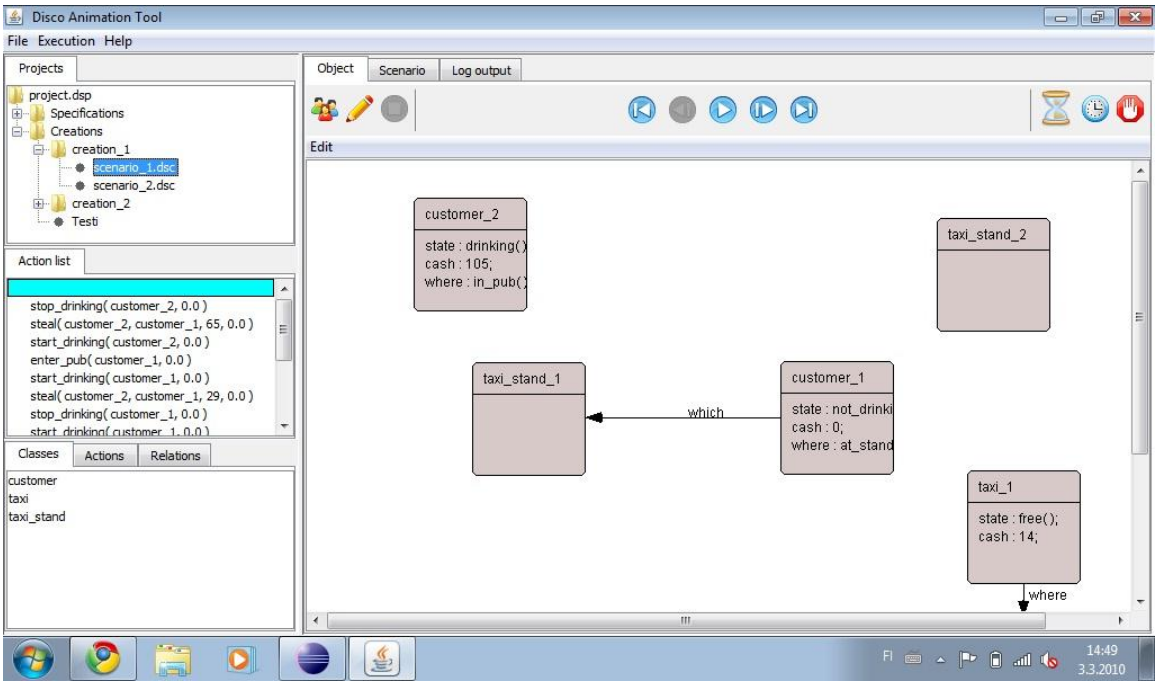
DisCo

Overview

DisCo (Distributed Co-operation) is a formal specification method for reactive systems. It incorporates a specification language, a methodology for developing specifications using the language, and tool support for the methodology. Currently the support tools include an animation facility for executing specifications, a tool for visualizing execution histories as scenarios, and a link to a mechanical theorem prover for verification. [http://disco.cs.tut.fi/General.html]

DisCo project concentrated on porting tool to Windows environment, refactoring it's codebase to use standard and modern Java frameworks and improving usability so that tool can be used by students in specification courses.

Complete product backlog can be found from the http://spreadsheets.google.com/pub?key=tiTAUJVAPFyuRgWfTjKG70g&single=true&gid=4&output=html

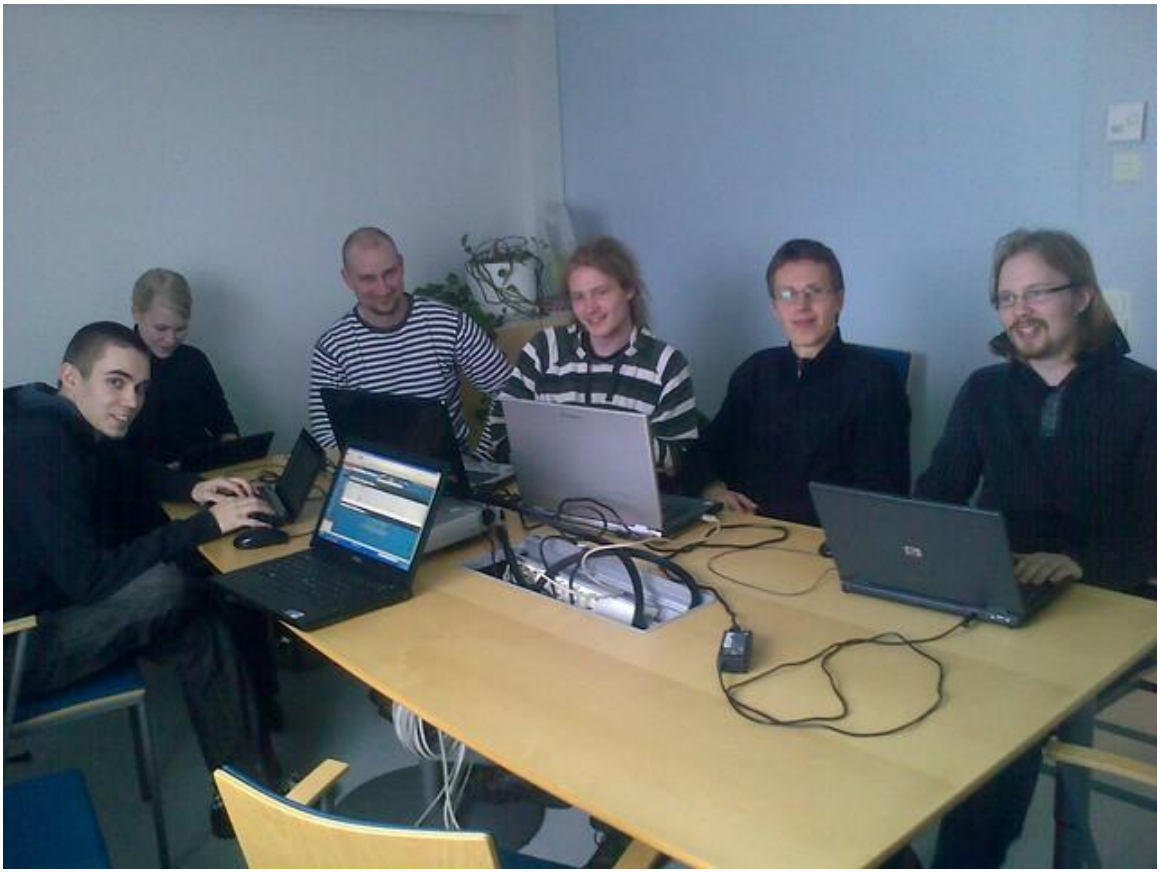


Picture 20. DisCo main window.

Organisation and management

- Jaakko Palokangas: Project manager focused on usability and facilitating Scrum
- Juuso Mäkinen: Project manager focused on technical aspects
- Sami Pekkola: Developer
- Mika Vallin: Developer
- Manteli Numminen: Developer

- Timi Vienola: Developer
- Jarkko Rinkinen: UX designer
- Laura Rosenqvist: UX designer



Picture 21. DisCo team. From left to right: Mika, Manteli, Timi, Sami, Audrius (developed further the Disco system) and Juuso. Jaakko took the picture. Laura and Jarkko are missing.

Methods and tools

- ScrumWorks Basic: Was used for product and sprint backlog management (project management in Scrum). Was clumsy to use and crashed frequently. For small projects looking for free backlog management tool I'd recommend JIRA with Greenhopper plug-in.
- Subversion: Defacto version control system for projects nowadays. Simple to use, it works and best thing is that this does not cost anything.
- DokuWiki: Most of the project documentation was stored to wiki. Wiki is the best way to handle any kind of project documentation with the exception of product and sprint backlog (if you have ever tried to prioritize wiki table you get my point).
- GoogleDocs: Were used for time tracking and backlog management after. ScrumWorks crashed for the last time. Not very usable for backlog management.
- NetBeans IDE: NetBeans was used most of team and most of the development

time until we had to merge code developed with Eclipse into it.
- Eclipse IDE: Eclipse was used in parallel with NetBeans (which was not the smartest thing to do). More configurable than Eclipse meaning also more difficult to use.
- Irssi: was the main tool for communication between face to face sessions. It is quicker than email and keeps discussion history.

Project phases and development model

We used modified Scrum

- Monthly sprints with a demo in the end of sprint as in vanilla Scrum
- Weekly scrums instead of daily
- Demoed and planned in two hours instead of one and half day, which was too less and you could see that on how poorly we reached sprint goals
- Skipped retrospectives because lack of time, which was a mistake retrospectively speaking.

Major project events were

- After September we had started project, we had tools running, project group has met and first product backlog and project and usability plan was done
- End of October we got DisCo running on Windows.
- End of March whole system was broken and we demoed piece of A5 in which we had our UI spec. ironically test plan was created during this time.
- Beginning of January things were not improved much and we started to have panic.
- Beginning of February we had first time something to actually demonstrate.
- Beginning of March we finally had a working system with project tree, basic disco functionality, refactored code and an installer.

Foreseen risks

- Project vision and scope was not clear: As anticipated the vision and scope of the project was unclear for the long time. This could have been avoided more intensive collaboration with the end users already in the beginning of project by whole team.
- Learning new technologies and skills took more time than anticipated: Neither of preventive measures worked well. On the other hand we took this into account in later sprints and then adjusted project scope accordingly.

Risks not foreseen

- DisCo legacy code was surprisingly spaghetti and poorly commented.
- Distributed development and of continuous face to face contact prevented team to gel properly making it less efficient and self organizing.
- Project management tool (ScrumWorks Basic) failed to work properly. There was also lot of problems with department's virtual environments which were shutdown regularly.
- Other school work took more time from DisCo project than anticipated. Silence did not mean that things were proceeding without problems.

Experiences

Main purpose of this project is to learn and that we did a lot. As in all realistic projects this was not solely joy ride but we had ups and downs. The most important thing was that we were finally able to produce working software. On the other hand we should have produced much more and already in earlier sprints.

What to do better next time?

See retrospectives at http://disco.cs.uta.fi/wiki/doku.php?id=retrospectives for more details but roughly

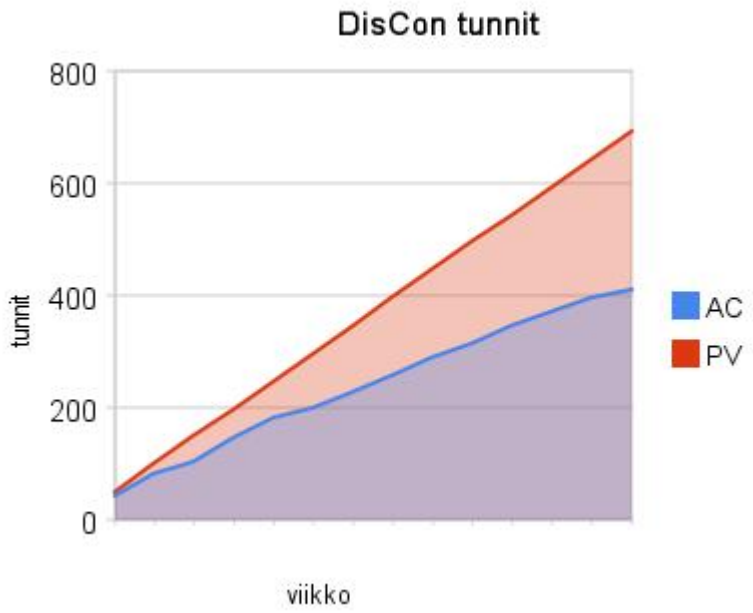
- Spend more time face to face with the especially in the beginning of project in order to get momentum and things rolling.
- Spend more time with the customer and end users so that we get deep understanding about what we are building.
- Do not skip retrospectives in hurry.
- Respect sprint goals and minimum target that is at least one working user story per sprint. React quickly if that is not reached.

Statistics

Team size	Dev. model	Start date	End date	Days	Hours
8	Scrum	15.09.09	16.03.10	182	870

Table 42: General project information.

.



Week	PP&Man	RS	Des	Code	Int&Test	Review	Repair	Study	Other	Total
38	19	8	0	0	0	0	0	11	4	42
39	17	0	0	4	0	0	0	12	8	40
40	15	0	0	0	0	1	0	2	4	21
41	17	0	4	8	0	0	0	6	10	45
42	9	0	0	9	0	0	0	5	14	36
43	6	0	1	4	0	0	0	5	2	18
44	13	0	0	7	0	3	0	4	3	29
45	13	0	0	3	0	0	2	6	4	28
46	15	0	0	11	0	0	0	2	4	32
47	9	0	1	9	0	0	0	4	5	27
48	14	0	4	9	0	0	0	2	3	32
49	13	0	1	5	0	0	0	3	4	25
50	15	0	0	5	0	0	0	2	4	25
51	6	0	0	5	0	0	1	0	1	13
52	0	0	0	0	0	0	0	0	0	0
53	0	0	0	0	0	0	0	0	0	0
1	12	0	0	6	0	0	0	1	2	21
2	20	0	0	27	0	0	0	5	5	57
3	14	0	0	29	3	0	0	2	4	51
4	13	0	0	29	0	0	0	1	5	47
5	24	0	0	19	0	2	0	0	5	50
6	17	0	0	23	0	0	1	0	6	47
7	20	0	0	23	0	0	0	2	9	54
8	15	0	0	48	0	0	1	2	7	73
9	21	0	0	28	0	1	2	0	9	61
										870

Table 43: Group effort by activity.

Number of requirements	Pages	User stories	UI screens	Database diagrams	Database tables
18	0	18	1	0	0

Table 44: Requirements and high-level design outcomes.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
0	0	16	0	0	1

Table 45: Design outcomes.

Language	JAVA
LOC	45885
SLOC	45885
Reused code	Lot of legacy code, project changes not counted
Reused and modified	Lot of legacy code, project changes not counted
Classes	627
Functions	Lot of legacy code, project changes not counted
Code revisions	212

Table 46: Codelines.

Document	Pages	Versions
Preliminary analysis	N/A, done to wiki	N/A, done to wiki
Project Plan	N/A, done to wiki	N/A, done to wiki
Usability analysis	N/A, done to wiki	N/A, done to wiki
Requirements specification	Not done	Not done
Design plan	Not done	Not done
User interface document	13	Many
Test plan	N/A, done to wiki	N/A, done to wiki
Test report	Not done	Not done
Usability test report	1	1
Final report	N/A, done to wiki	N/A, done to wiki
Project's story	8	2
Weekly reports	N/A, done to wiki	N/A, done to wiki

Table 47: Documents.

Content Manager Mobile

Overview

The project's aim was to expand Ambientia Oy's Content Manager System to allow creation and management of mobile web pages. In the course of our project, we designed a mobile web page for Evira, the Finnish Food Safety Authority and outlined design questions specific for mobile web pages. An important part of the development process was to document it so our work could be utilized by Ambientia in its future projects.

In addition to page templates and design, we did much work on the internals of the product. The main expansion to the CMS was an extension module to handle certain kind of information and that can be used from page template code. Furthermore we wrote smaller pieces of software, such as parsers and the administration interface for the extension module.

Both the extension module and the Evira's mobile web pages have now been completed. The project will be turned over to Ambientia as the project work course ends and will probably be published some time this year.

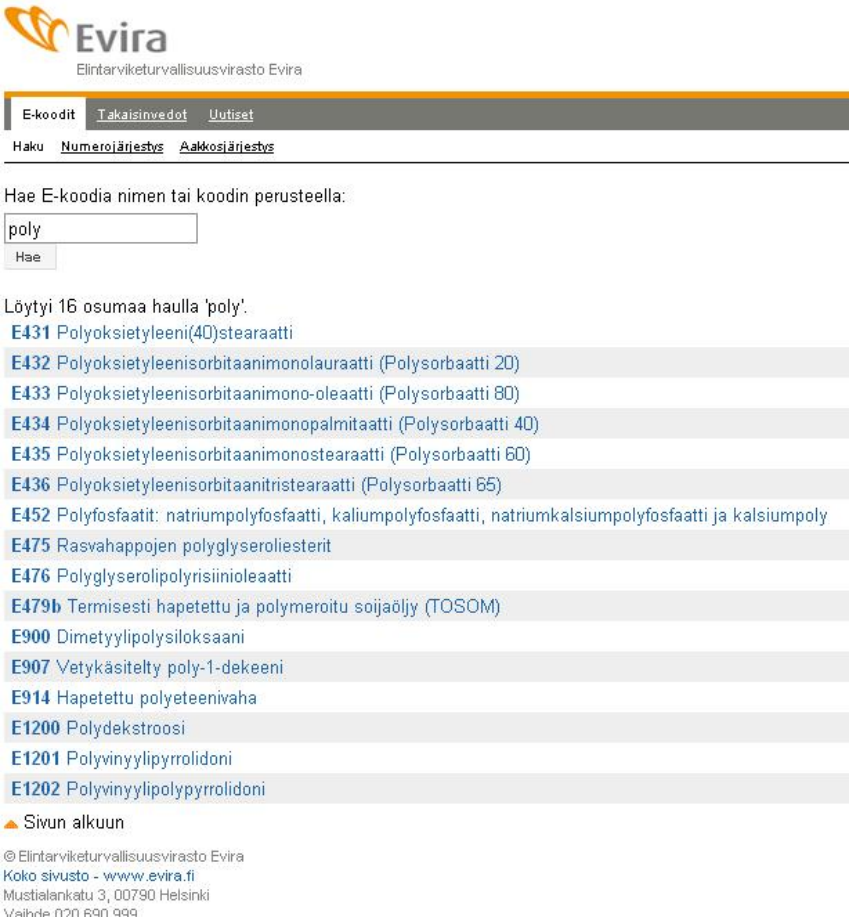


Figure 22. Main screen of the application.

Organisation and management

Project members and their roles:

- Jenni Haataja, project manager
- Niina Ojala, project manger
- Ville Mäkelä, programmer
- Jari Laitinen, programmer
- Ilkka Virolainen, programmer
- Evgeni Pajunen, programmer
- Anu Leppälampi, usability expert

Niina and Jenni were the project managers of this project. Niina was responsible more on the technical issues such as writing installation guidelines and Jira-tasks. Jenni was more responsible on organizing issues such as making room reservation for the weekly meetings, writing weekly reports and updating sprint backlog.

Picture 23. The project team.

Methods and tools

Content Manager 6.0 - Ambientia's main product. Not available for the public.

Confluence - A web-based corporate wiki written in Java and mainly used in corporate environments. Used by Ambientia and the project team to store all the information related to the project. Easy to use. Free only to non-profit organizations and open source projects.

Jira - A proprietary enterprise software, commonly used for bug tracking, issue tracking and project management. A bit hard to use, but handy for project managers to assign tasks and follow the work of project members. Jira is provided free to open source projects, and organizations that are non-profit, non-government, non-academic, non-commercial, non-political, and secular.

OpenVPN - A program needed to connect to Ambientia's server in order to use Confluence, Jira and Content Manager. Free.

Eclipse - A software development environment used to program Content Manager Mobile. Useful and multifaceted, but heavy. Free.

Skype - An instant messaging program used by the project team to communicate with each other daily. Easy to use, free.

Project phases and development model

Project's development model was modified scrum. The project was divided into 5 sprints, which lasted from 3 to 5 weeks. Instead of daily scum meetings, there were weekly scrum meetings. They lasted about 1 to 2 hours and only project group participated in these meetings. First everybody told what they had done, what they were going to do next and did they had any problem. After that the project group discussed about some issues which needed attention. At the end of every sprint there was a scrum review meeting combined with scrum planning meeting. The customer as well as the project course leader participated in these meetings together with project group. First project group demonstrated what they had done. Then the customer told what he wanted to be done next. After the customer and the course leader had left the group discussed what had went well, what did not went well and what could be done better in previous sprint.

First sprint lasted 5 weeks. In this sprint project group installed developing environment in their PC's and they made mock model about the web page they were going to implement. Mock-model is like prototype about the web page which don't have full functional, but the structure of the page is easily seen. The second sprint lasted 4 weeks. Tasks in this sprint were creating use cases, creating new branch for mobile web pages in content manager, creating news and product defects pages into content manager and developers' diary was started. The third sprint lasted 3 weeks. Task of this sprint were creating paging and navigation for the mobile pages, creating a video about mock-model and how it can be used, fixing styles, creating a template for adding e-codes in admin side and creating public page templates for e-codes. The fourth sprint lasted 4 weeks. Tasks in this sprint were finishing the coding. The fifth sprint lasted 5 weeks. Tasks in this print were mainly testing and bug fixing.

Milestones

- Presenting project in project management course on 16.9.2009
- Review of the preliminary document on 23.9.2009
- Review of the project planning document on 9.10.2009
- Presenting mock model to customer on 15.10.201
- Sprint review of the first sprint were on 28.10.2009
- Sprint review of the second sprint were on 25.11.2009

- Presenting project in project course on 9.12.2009
- Sprint review of the third sprint were on 16.12.2009
- Sprint review of the fourth sprint were on 4.2.2010
- Sprint review of the fifth sprint going to be on 4.3.2010
- Review of the whole course is going to be on 8.3.2010

Sprint backlog for the first sprint:

- Everybody installs the developing tools
- Mock-model of the web pages
- Presenting mock model to customer
- Preliminary plan about how the CMM could be technically implemented

Sprint backlog for the second sprint:

- Create a branch for the content manager mobile to the content manager
- Create news and product defects page templates in content manager
- Use cases
- Correcting mock-model
- Starting developer's diary

Sprint backlog for the third sprint:

- Updating styles of the mobile pages
- Finalizing news and product defect pages
- Navigation for the pages
- Paging for the news and product defect pages
- E-codes page templates
- Template for adding e-codes in admin pages
- Video for using mock-model

Sprint backlog for the fourth sprint:

- Inserting e-codes to the database
- Implementing logic for the admin pages
- Implementing logic for the public pages
- Implementing search for the e-codes
- Converting pictures into links
- Converting buttons into links
- Planning usability test
- Meeting with Evira

Sprint backlog for the fifth sprint:

- Finishing e-code module
- Fixing bugs
- Helding usability test
- Customer meeting with Evira
- Creating project CD
- Creating technical document
- Presenting project

One project group member quit the course. This was not foreseen at the beginning of the course, because everybody seemed to be very interested and motivated about the course.

Experiences

All group members had only good things to say about working with the group. Group spirit and atmosphere were excellent.

All group members thought that the project was well lead. Project managers informed project group about their tasks and meeting times on time. And overall the management was good.

Project managers didn't have previous experience about how to handle a situation when one of the group members constantly didn't do his / hers job on time.

Group members took the initiative to do tasks on their own and they did a good job after the project managers had shown the right direction.

Statistics

Team size Dev. model Start date End date Days Hours

2+4+1 Scrum 2.9.2009 8.3.2010 188 1165

Table 48: General project information.

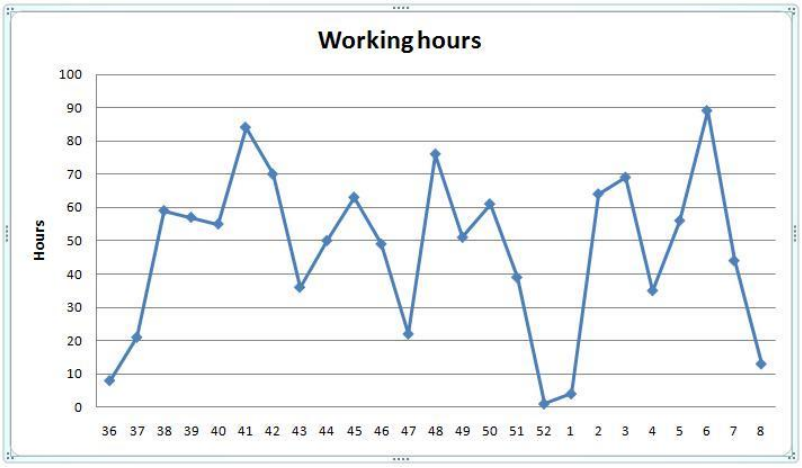


Figure 24: Working hours.

Activity	Planning and manage ment	Require-ments speci-fication	De-sign	Code	Integra-tion and testing	Re-views	Re-pair	Study	Other	Total
Hours	421.5	11	58.5	240.5	10.5	38.5	24.5	80.5	279.5	1165
%	36.2	0.9	5	20.6	0.9	3.3	2.1	6.9	24	100

Table 49: Group effort by activity.

Number of requirements	Pages	User-cases	UI Screens	Database diagrams	Database tables
		9			

Table 50: Design outcomes.

Document	Pages	Versions
Preliminary analysis	7	5
Project plan	22	21
Usability analysis	2	1
Requirements specifications	5	1
Design plan	*	5
User interface document	-	-
Test plan	**	-
Test report	**	-
Usability test report	15	1
Final report	17	1
Project's story	7	1
Weekly reports	51	1
Technical report	6	1

Table 51: Documents.

* Mock model was our design plan and we made 5 different versions of it. That was the basis of our coding.

** We didn’t have test plan or test report, because we tested iteratively all the way through the project.

Language	Java EE, Velocity, HTML, CSS
LOC	2594
SLOC	~2000
Classes	20
Functions	100

Table 52: Codelines.

MediaBank

Overview

MediaBank is a web-based service which allows users to easily upload and share their media files. It is designed to be used with touchscreen devices, such as the Apple iPhone, but any touchscreen device with a JavaScript-compliant web browser should suffice. Users can upload and share their media files within MediaBank with great ease. Users are also able to perform searches on 3rd party services, such as YouTube and Flickr. MediaBank service acts as a hub that allows the user to easily access content from one's private content storage as well as from popular social media services.

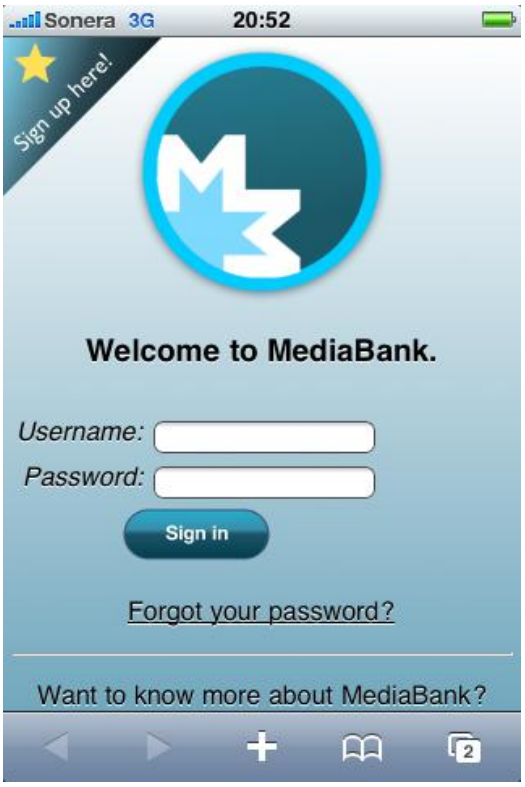


Illustration 25: MediaBank's front page.

Illustration 26: Gallery shows user's files.

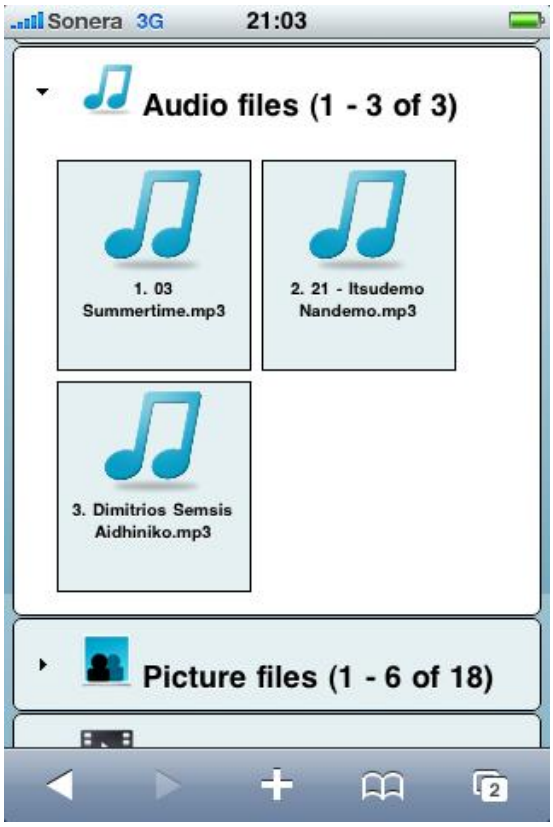


Illustration 27:Closer look at gallery and individual files.

Organisation and management

The client for the MediaBank-project was TeliaSonera. Mr. Rami Lehtonen has been representing TeliaSonera during the project. Also during development the project the group was collaborating with Demola. Demola provided the project with an SVN-repository, wiki and other necessary tools that the project demanded. Demola is also the party which has paired the client and the project team for this project. In addition, Demola has also offered it's premises to be used by the team for necessary purposes. Mr. Ville Kairamo has been representing Demola during the project.

Project members

- Matti Pesonen (Project leader)
- Mikko Tillikainen (Project leader)
- Marko Haarni (Project member)
- Atte Karhunen (Project member)
- Markus Sairanen (Project member)
- Jussi-Matti Salmela (Project member)
- Heikki Salo (Project member)
- Anu Leppälampi (Project and Usability team member)

Illustration 28: From left to right, top to bottom: Markus, Anu, Jussi-Matti, Matti, Heikki, Mikko, Atte, Marko.

Methods and tools

During the course of the project, the following tools have been in use:

- Eclipse
- Dokuwiki
- MySQL
- JavaScript, jQuery

Eclipse has been a great help during the project with it's auto-completion and SVN-features, among others.

Dokuwiki was used to create documents and write down discussed matters during project meetings.

jQuery offered many easy-to-use and rather aesthetically pleasing UI-elements for the system's demands.

Project phases and development model

Development model for the project was a modified Scrum. It was impossible to have daily meetings, so instead meeting were held weekly.

Sprints during the course usually lasted 2 to 3 weeks. During the course there were reviews after every sprint with the client.

- 25.9.2009 Sprint 1 review
- 9.10.2009 Project plan review
- 27.10.2009 Sprint 2 review

- 19.11.2009 Sprint 3 review
- 10.12.2009 Sprint 4 review
- 27.1.2010 Sprint 5 review
- 24.2.2010 Final review

Experiences

The starting point for the project was not very promising. The team was supposed to further develop an already existing system, which had been created earlier by team consisting of one person. During the development of the original version, the specifications had obviously changed a few times, and the system overall seemed somewhat rushed. Existing code had not been commented or documented. This meant that after several weeks of asking for documentation, the existing system was scrapped and the project was started again from scratch.

Statistics

Team size	Dev. Model	Start date	End date	Days	Hours
2+5+1	Modified SCRUM	07/09/09	07/03/10	182	1265

Table 53: General project information.

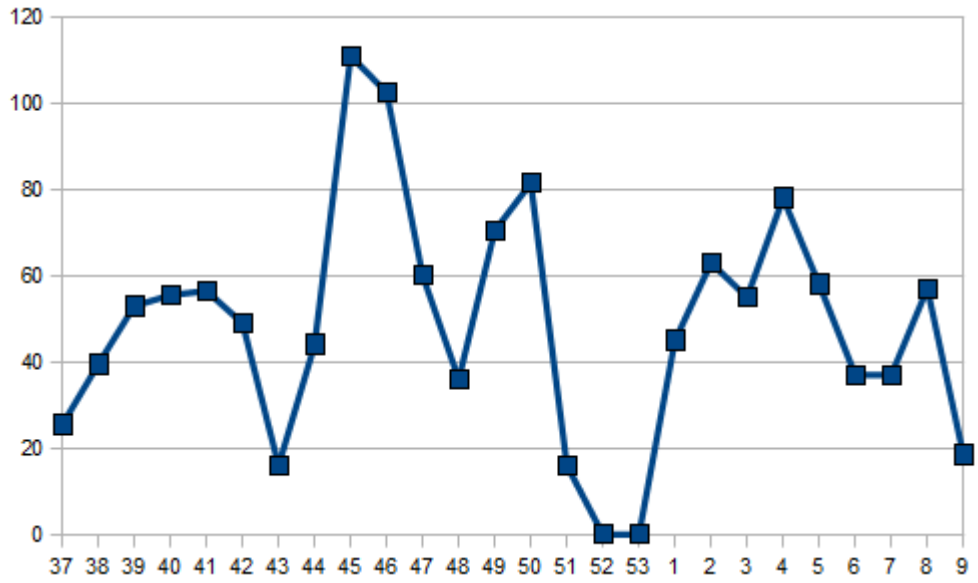


Illustration 29: Working hours per week

Activity	Planning and management	Req. specification	Design	Code	Integration and testing	Reviews	Repair	Study	Other	Total
Hours	389.5	46	141.5	180	13.5	43	35.5	225.5	190	1265

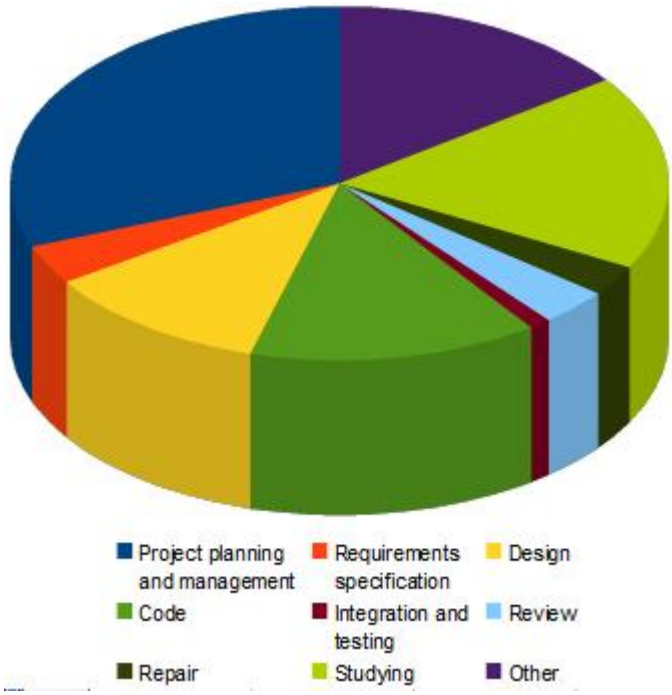


Illustration 30: Group effort by activity.

.

Use-cases	UI-screens	Database diagrams	Database tables	Overview diagrams	Other diagrams
12	109	2	13	1	1

Table 54: Design outcomes.

Document	Pages
Preliminary analysis	6
Project Plan	25
Usability analysis	14
Requirements specification	10*
Design plan	5*
User interface document	34
Test plan	13*

Final report	X
Project's story	X
Weekly reports	23

Table 55: Documents.

Type	Files	LOC	LOC per file
*.java	29 (41.4%)	4320 (69.8%	148.9
*.jsp	26 (37.1%)	1131 (18.3%)	43.5
*.xml	9 (12.9%	530 (8.6%)	58.8
*.css	4 (5.7%)	187 (3.0%)	46.7
*.properties	1 (1.4%)	17 (0.3%)	17.0
Others	1 (1.4%)	3 (0.0%)	3.0
Totals	70 (100.0%)	6188 (100.0%)	88.4

Table 56: Codelines.

Majava2

Overview

Project Majava2 is an online competition platform for the Majava computer science competition. This project continues previous work done by the last year's project work course and most of the tasks involved realizing new features and fixing old design faults.

The technologies to be used were not for us to decide as we continued working on an existing code base. This was a big problem as our team believed that they could have done the project a lot better with the tools of their choosing.

The project was done in part with the help of Toijala Nokia Research Center and the Päivölä math high-school students working there.

Organisation and management

Picture of the group. Upper row from left to right: Olavi, Antti, Tuomas and Juhani. Second row from left to right: Taina, Minna and Cindy. Next to the group photo is Ville.

- Project managers: Antti Sand and Olavi Karppi
- Project members: Taina Lempiäinen, Juhani Linna, Cindy Oguji, Ville Pakkanen, Tuomas Pellonperä, Minna Heinonen (usability team)

Only a small part of the project group had any interest or skills in programming with the Ruby programming language and the Ruby on Rails programming framework. Most programming tasks and the maintenance of the Debian Linux running virtual server fell to Tuomas Pellonperä. Taina Lempiäinen and Cindy Oguji also did a lot of programming work. Juhani Linna and Ville Pakkanen were more focused on the usability factors and other tasks like entering and modifying the questions database. Minna Heinonen was our usability team member, but she took some part in the questions database work also.

The Päivölä team focused mainly on the administrative side of the platform and their work is not included (in any greater detail) in this project.

Methods and tools

Main tool used in the project was MediaWiki which was used for storing schedules, task lists and documents. Wiki-software was proven to be a good solution for common collaboration tool. For task management some more specialized tool would probably been better choice as task handling started soon to be pretty laborious.

The software written in project employed Ruby on Rails framework. Development was done using Aptana Studio and normal text editors such as Notepad++.

The Aptana Studio for Ruby on Rails proved to be a pretty good IDE. It housed some powerful features for virtual environments and debugging.

The Ruby on Rails framework didn’t get the groups favor and might not be the preferred choice for any new projects.

Project phases and development model

Majava2 used iterative development model. When starting the project the idea was to use modified Scrum. Modifications were needed as for example the daily meetings specified in Scrum were not something that could have been done in a school project.

Project had 7 iterations. First iteration was used for just plain planning. Project Plan was written during first iteration. First three iterations were about one month in length. Later iterations were shortened to about two weeks each as it was noticed that one month iterations were too long for project's needs.

One of the challenges that was noticed pretty early was that almost all project members were working and that many had full time jobs, so the time resources which could be used were perhaps little more limited than maybe in some other project groups. Second big challenge was met late in the project when the project's virtual server had to be taken down due to someone's hacking attempt.

Unforeseen problems

In the beginning of the project, after some weeks of work, our group received an extra member. Some time later though this person decided to quit the course and this affected the distribution of the workload twice.

In the middle of the course another group member moved to another city and could not attend any of meetings anymore. These problems are common in projects though.

Due to the hacking attempt we suffered many weeks of downtime and in addition to all the extra work that brought, the project group couldn’t do much for the project. The Wiki, the documents and the source code version control were inaccessible for a long time at the end of the project which effectively screwed all plans and deadlines. This was a Force Major issue as we had no means of acquiring a secondary server for our project.

Experiences

Hardest part seemed to be to create solid decisions and to follow them. Iteration deadlines were often missed. For future more detailed specifications and closer follow-up on the tasks under work could provide better results. Also, if new technologies are used then the basics of them should be learned quickly in the beginning of the project.

Motivating the team to upkeep a consistent workflow through the entire project was difficult as most of the group (if not all) were already working and studying simultaneously, which left little time and energy to put in this project.

Statistics

Team size	Dev. model	Start date	End data	Days	Hours
2+5+1	Iterative	24.09.09	25.03.10	~180	1008,5

Table 57: General project information.

Acti-vity	Plan. and mana geme nt	Req. speci fica-tion.	Desi gn	Code	Integ-ration and testing	Re-vie ws	Re-pair	Study	Ot-her	Total
Hours	247	2	40	245	18	18	12,5	152	274	1008,5
%	24	0,2	4	24	1,8	1,8	1,2	15	27	100
Usabi-lity	0	0	20	0	0	0	0	1,5	30	51,5
Total	235	2	60	254	17	18	11	153,5	268	**1060**

Table 58: Group effort by activity.

Number of requirements	Pages	Use-cases	UI screens	Database diagrams	Database tables
-	-	-	18	2	14

Table 59: Requirements and high-level design outcomes.

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

Table 60: Design outcomes.

Document	Pages	Versions
Preliminary analysis	7	2
Project Plan	15	wiki
Usability analysis	10	7
Requirements specification (user stories, backlogs)	10	wiki
Design plan	3	wiki
User interface document	17	5
Test plan	18	5
Test report	-	-
Usability test report	-	
Final report	9	1
Project's story	5	3
Weekly reports	24	-

Table 61: Documents.

Language	Ruby
LOC	3504
SLOC	3504
Reused code	-
Reused and modified	-
Classes	-
Functions	-
Code revisions	230

Table 62: Codelines.

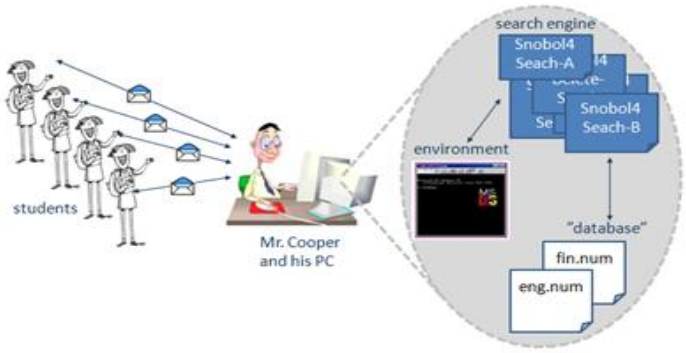
TamBiC Project Story

Overview

The client and the primus motor of the TamBiC project was Mr. Robert Cooper from the Tampere University School of Modern Languages and Translations Studies. The project was aimed at creating a web application that would serve as a search engine for the existing collection of texts, i.e. a 'corpus'. The web application was also to provide a user interface for editing texts and adding new texts in the existing text collection.

Over the years, Mr. Cooper has carefully selected and gathered a collection of texts in both English and Finnish language. Accompanied by their corresponding translation texts, these texts form two separate corpora - one with originally English texts with their Finnish translations, and one with originally Finnish texts with their English translations - which are stored in two large text files in Mr. Cooper's personal computer. The English philology students in Tampere University use this "database" to gather research material, for example on translation practices concerning certain words or phrases (like how word "play" is translated in Finnish in different contexts), for their Master's Thesis works.

The old system worked so that students contact Mr. Cooper personally, asking for material on certain words or phrases. In order to perform different kind of searches in the 2 text files, Mr. Cooper has used his own Snobol4-programs in MS-DOS environment. After Mr. Cooper has made the searches, then e-mails the findings back to the students for them to analyze.



Picture 31: old TamBiC system.

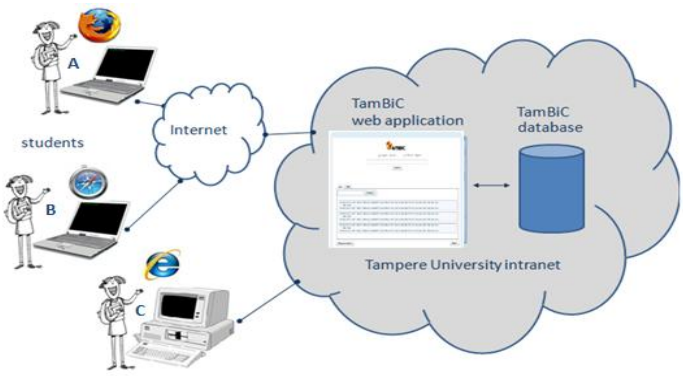
As all the searches in the text files are done on command line basis, and the programs input and output .txt files, the risk of accidentally over-writing the actual corpora files is always present. Also, the Snobol4-programs are a bit difficult to use for modern students, who are more used to using graphical interfaces than MS-DOS environment.

The new TamBiC system

As Mr. Cooper will be retiring in a few years, he would like to leave the corpora he has collected as his legacy to his department. However he wanted the TamBiC corpus system to be updated so that it would be enticing, exiting and easy to use for the future students - as he would not be there to guide them using it. For that purpose, an on-line search engine would be needed.

- TamBiC project mission: **transform the existing corpus system into a web application**
- TamBiC project goals:
  - ✓ Make the bi-lingual corpora **easily approachable** by reducing the complexity in the way sophisticated searches are done.
  - ✓ Provide users with a **modern looking**, **user friendly and intuitive** GUI.
  - ✓ Allow a **large number of users to access** the database simultaneously for searching material, while securing the database content to be editable by one key user (i.e. the administrator) only.
  - ✓ **Empower** students and staff by providing them with means to **independently use the corpus on-line** to perform the searches and language analysis in their personal work.
  - ✓ Make **operations** in the corpus database **fast and reliable**.

In the new TamBiC system, the students can access the corpus database via the Tampere University intranet. The corpus search engine is a web application, available via any web browser.



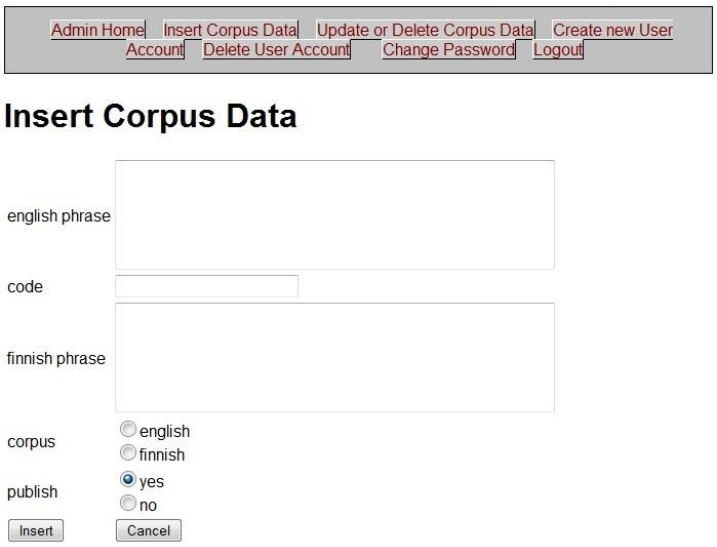
Picture 32: new TamBiC system.

The provided new TamBiC system consists of 3 parts:

1. **SQL database holding the contents of the two corpora.** Features:
   - Provides efficient methods for searches based on string pattern matching
   - Provides easy methods for inserting and editing the corpora contents

2. **Administrative web application for developing the database**. Features:
   - Web form for making searches on rows (by text code)
   - Web form for editing and deleting rows
   - Web form for adding new data to the database

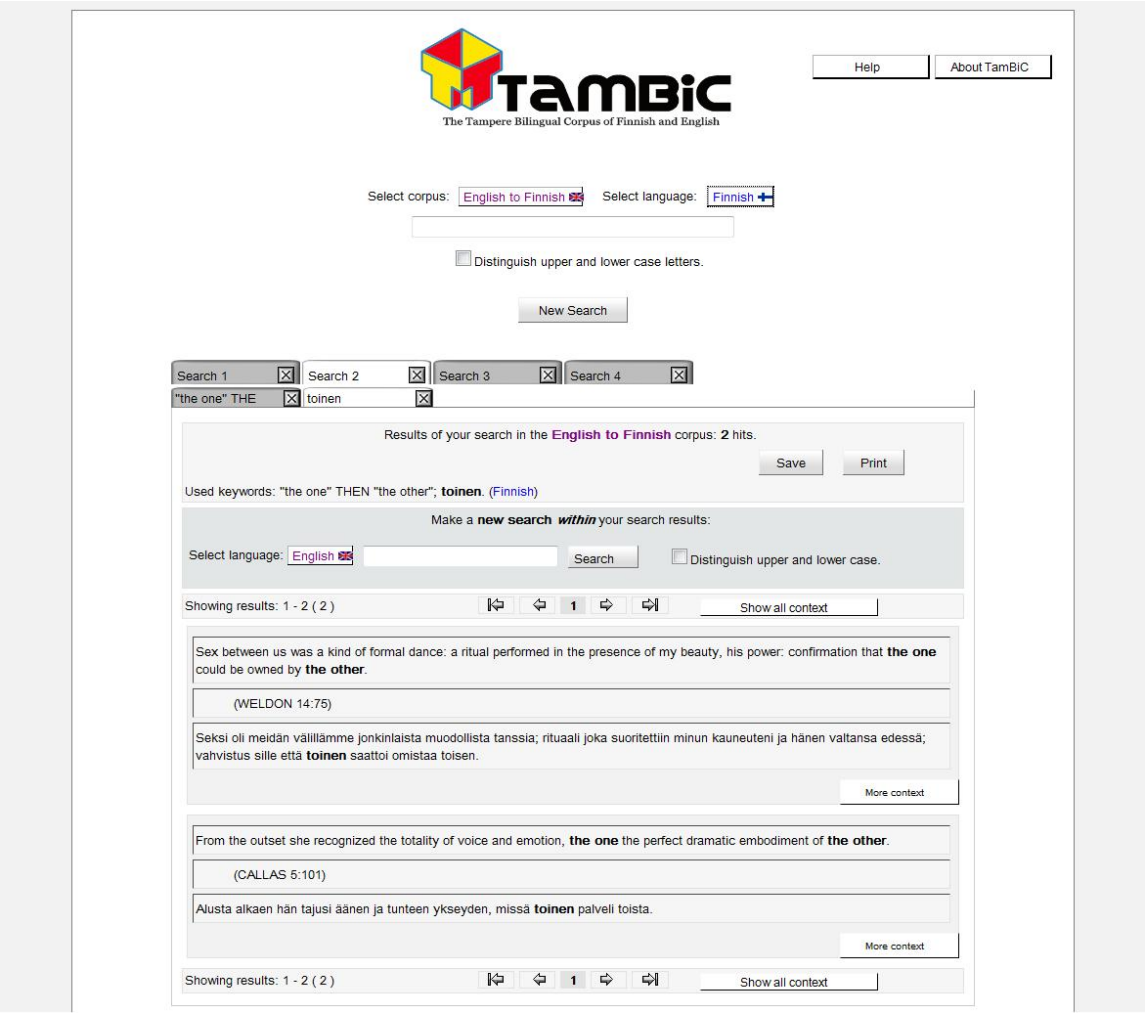
3. **Web application for making the searches in corpora.** Features:
    - Basic searching, where user needs to select corpus and language (English/Finnish)
    - Making a new search within your current search results (i.e. refining results)
    - Versatile search options:
        - Word matching w/o wildcard (e.g. play, play*, *play, play*, *play*, …)
        - Phrase matching w/o wildcards (e.g. “the one”, “the *ones”, “the * one”, …)
        - Case in/sensitivity: turn On/Off (e.g. cat, Cat, CAT, …)
        - AND, OR and NOT syntax (e.g. cat AND dog, cat NOT dog, cat OR dog, …)
        - Showing more context, which displays the next and previous sentence in the text
        - THEN syntax , which finds sentences where words are in the given order
    - Graphical UI, where the results are shown with hit counting and matching words being bolded in the search results to provide an easy verification of the results
    - Having several searches open at the same time, since every results is put in a new tabs.
    - Saving results (in a .rtf file) w/o sentence context being saves, as well
    - Printing results (from a new .html page) w/o sentence context being printed, as well
    - “Help” page, which contains the user manual on how to use the system
    - “About TamBiC” page, where user can find more information on e.g. the text sources.



Picture 33: screenshot from “insert data” from in Admin tool

Project organization and management

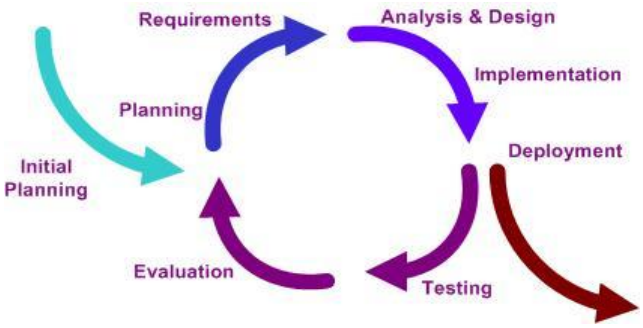
- Project members and their roles.
    - Reeta Karjalainen - Project Manager
    - Georgina Monjaraz Gomez - Project Manager
    - Harri Mustonen – Developer
    - Pekka Ihalainen – Developer
    - Simone Strasser – Developer
    - Xiaozhou Li – Developer
    - Juhani Vainio - Usability Expert



Picture34: TamBiC search engine GUI

Iterative development model

The project followed an iterative development model, in which all development work was carried out in several iterative rounds, each lasting some 2-4 weeks depending on the iteration goals.

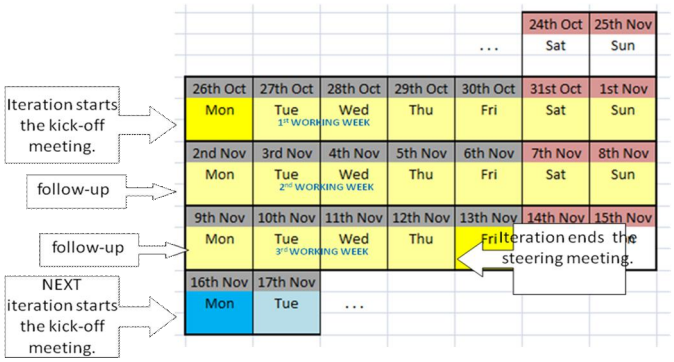


Picture 35: iterative development model

Each iterative round starts with planning the goals and contents of the iteration. The

team discussed the requirements and work load estimates with the client and selects in co-operation a few new features that are to be implemented. The needed implementation work and testing is then planned and assigned to developers, taking into account other possible tasks - such as repair work from previous iteration - that needs to be carried out, too.

Next the project team will implement the features, and continue to integrate the new software into a new running system, and tests the implementation, as well. At the end of the iteration round, the results of the iterative work is evaluated with the client. In the iterative development model, the functional capabilities of the system will increase in every iteration round, and after several rounds, the software will finally meet all the set requirements.



Picture 36: example on how to organize an iteration

The project work very well with this model as each steering meeting with the client we discuss the features to be implemented and performed the changes needed from the last iteration week. Also meetings with the course directory, Timo Poranen, were very useful to show progress and get advice.

About daily working and practices

In the project team, the different development tasks were assigned in the planning meeting also in the fly, according to the current needs of the project. Naturally both personal interest and skills were taken into account. As a result, some people focused only on one individual task area due to the limitation of their availability, while some developer work doing pair programming.

Management tasks were divided half and half on weekly basis, so one manager could do the weekly management tasks for that week, including the weekly report, while the other have time to do some other assigned team tasks. In meetings and presentation work was always done by the two managers on same proportion. The usability expert assigned to the project participated eagerly every project meeting, and he kept a good record of agreed UI design guidelines and the client's needs. He also produced the usability analysis and managed arrangements for the usability testing sessions.

At the very beginning of the project, two of the originally assigned 6 developers quit the project, leaving the project with 4 developers. Luckily the project work had not event properly started when this happened, so that didn't really jeopardize the project

work. However, lack of resources did have an effect on the project results. Also one team member was working full-time during the course, so we could only get a 60% attendance from that developer. In the end all team members, including project management and usability personnel, took part in all project activities such as testing, coding and documenting.

Picture 37: Team TamBiC in picture: first row (from left to right): Georgina, Reeta and Juhani. Second row (from left to right): Xiaozhou, Pekka, Harri and Simone.

Methods and tools

Project documentation was done in team's PBWorks Wiki pages (see http://tambic.pbworks.com/), where you can also upload documents and images. The Wiki proved very useful as it allows everyone to see the current latest information and access documents easily while keeping the work area private and organized. PBWorks Wiki doesn't need any installation and is free to use - at least with the basic options; for more advanced support there's a monthly fee. The wiki user interface was easy to understand and you can also edit the wiki pages in HTML mode, which was useful for creating some files in the project work too (like Help for the search engine).

Communication of the team was mainly handled via the weekly meetings and emailing. Occasional phone calls Messenger were also employed. So far, the team considers that the wiki site and emailing were the most efficient methods for assigning tasks and leveraging project information.

The TamBiC search engine and admin tool were created using JavaScript and PHP scripts upon web pages implemented using HTML and CSS. The search engine UI and its dynamics are implemented in JavaScript - or actually, they were written in Java, taking advantage of the Google Web Toolkit, which renders the written Java code into executable JavaScript.

Google Web Toolkit (GWT) is a development toolkit for building and optimizing complex browser-based applications. The software is completely free and allows you to write AJAX applications in Java and then compile the source to JavaScript that can

run across all browsers. While installed in Eclipse as a plug-in you are allowed to inspect variables, set breakpoints, and utilize all the other debugger tools available for Java. While GWT is versatile, using it in a complicated UI project requires a structured development environment, like the Eclipse we used. Working with GWT produces results quickly in running code, but due to the lack of general "framework" in JavaScript, maintaining and editing the code files proved a bit tedious.

PHP scripting was used for creating dynamic web page content in the Admin tool web page forms. For that usage, PHP code was embedded into an HTML source document and then interpreted by a web server, which generates the web page document. PHP was also used to execute SQL queries in the database and to create printable/savable result files. As PHP can be used for command-line scripting and client-side, it also worked for us as the primarily filter in the search engine, taking input from a file or DB and outputs another stream of data. PHP is free to use, and it's released under the PHP License.

The actual database was created in PosgreSQL, an open-source Object-Relational DBMS supporting almost all SQL constructs.

Project phases

The following table describes the project phases; it lists all major and minor milestones/checkpoints including contents of the inspections with dates, as well as the documents delivered in each phase. The list of iterations with their content is also showed. The "Nbr" refers to the iteration round number.

Nbr	Dates	Description	Products, Meetings and Dead-lines
0	weeks 37-38 **= 2 weeks**	*Project work is started, and project group tries to organize itself and figure out what to do…*	• 10.9. Project members selected. • 4.9. First meeting with the client. • 16.9. Project idea presented to all other project managers in course.
1	weeks 39-42 **= 4 weeks**	*Initial project planning.* *First drafts on how the product should work and look like.* *Actual coding work is started.* ***TARGET: 1st working version of the software, with all the components in place.***	Weekly review meeting with team every Monday. • 25.9. Preliminary Analysis review. • 28.9. 1st draft of Usability Analysis doc. • 9.10. 1st draft of Project Plan sent out NOTE: writing of these documents is started in TamBiC forum: Requirement Spec, UI Design, Test Plan and Implementation Plan.
	week 43	***Autumn holiday.*** *Individual tasks,* *updating document,*	• 19.10. Steering Meeting of the 1st iteration: client is shown how the search tool will look and work like. • 19.10 Project Plan review meeting on week 43

		Preparing for 2nd iteration.	
2	weeks 44-46 **= 3 weeks**	Achievements: • a real, running **1st versions of the TamBiC search engine UI** is available • **Basic initial search is working** • Print results on screen is working, also counting the number of hits. Case sensitivity can be considered or disregarded.	• 26.10. Kick-off meeting to start the 2nd iteration. • 28.10. Meeting with the client to clarify the search requirements. • 6.11. Send out Personal Report I. • 13.11. Steering Meeting to end the 2nd iteration.
3	weeks 47-49 **= 3 weeks**	*…more features added to the search tool:* • **AND and OR syntax working** • **"Delete searches" (i.e. NOT operand) working** *Database is also in place, as well as the UT database with limited amount of data for easy result verification.* *Start planning usability tests with real users (for weeks 1-3).*	• Monday 16.11 Kick-off meeting to start the 3rd iteration • Friday 4.12 Steering Meeting to end the 3rd iteration.
4	weeks50-51 =**2 weeks**	*…more features added to the search tool:* • **Punctuation sensitivity** to be consider (n't) **Showing more context is working.** *Problems with "refined search" are delaying all other work…*	• Monday 7.12 Kick-off meeting to start the 4th iteration • **9.12. Product Presentations.** • Friday 18.12 Steering Meeting to end the 4th iteration.
	weeks 52-53	***Christmas holiday.*** *Individual tasks…* *Problems with "refined search" are delaying all other work…*	
5	weeks 1-3	*Help menus added.*	• 6.1. Send out Personal Report II.

	= 3 weeks	*Problems with "refined search" are delaying all other work…*	• 22.2 Steering Meeting with the client for end of iteration.
6	weeks 4-6 **= 3 weeks**	***Whole search logic is re-written*** *to provide also the refined search functionality.* ***All search engine features are being maturated…*** *Help menus are written. User Manual is written. Usability Testing.*	• **Implementation should be READY by 15.2!** …after that date, only bug fixing is allowed. • 5.2 Pilot usability testing. • 8.2 Steering Meeting with the client is shown the project almost finished. • 10.2 Usability testing with volunteers.
7	weeks 7-8 **= 2 weeks**	*All search engine features are being maturated…* ***THEN syntax is added.*** *New search requirements are implemented (mostly just fine - tuning the way features work)* ***Admin tool is developed.*** *User manual and Help are revised.*	o Usability findings are review and corrections are done. o Heavy search engine testing is on-going… o Admin tool testing started
8	Week 9	***Final Presentation.*** *All project products (document etc) are finished.*	• **Final presentations are given**. • Testing is completed. • All UI findings are corrected now.
9	Week 10	***Save and Print functionality implemented.*** ***Final installations are done.*** *All project products (document etc) are finish and showed to the client*	o 9.3. Final Team meeting o 12.3. **Final feedback meeting** with the client. o 14.3. Send out Personal Report III.
10	Week 11	End of the project!!	15.3 **Final feedback meeting** with the Timo Poranen. Return all project documentation and project CD.

Foreseen and unforeseen problems and risks the team was faced with:

- We lost 2 team members at the beginning so more work was assigned to developers. Also, switching tasks, e.g. in case someone wasn't able to carry out his/her tasks, was sometimes difficult, since there was no-one free to take over.
- Inexperience in programming cause a lot of problems: we needed to re-design the whole main script again, making use lose some +100 hours of work altogether…
- Due to staffing shortage and inexperience, some features were implemented rather late, and then there was even less time to fix all the detected usability issues and add all those fine-tunings the client desired to have…
- The usability testing was done a lot later that originally planned, since the program was not ready to be used for testing…
- Usability test show a lot of issues that had to be corrected in the UI.
- UI changes were not very easy to do; a lot of time spends here.
- Some of the expected features were left out due to time.
- Problems with Scandinavian characters all the time (also, the PostgreSQL regular expression functions, such as word boundary detection, did not support Scandinavian letters -> a lot of hand coding was required).
- Better understanding of the project at earlier phase might have helped us to better design code structure and select appropriate tools…
- The whole project should be been started with usability testing some paper models of the UI. Now a lot of time was spend on developing a UI that failed to meet the user's expectations. We should have also taken into account that our client, who has several years of experience on the corpus search engines, may have different perspective to the necessary features that an average student user.

Experiences

In the end, the team managed to handle all problems and delivered client the agreed SW - with even some last minute improvement requests incorporated. On the positive side,

- A lot of knowledge gain from doing the project.
- Problems were solved in the main script but a lot of work was redone.
- Project is functional and reaches expectations from the client.
- The UI is friendly to use and understandable thanks to the usability testing.
- Large number of users will be able to access to the application.
- The group managed to work in a good and responsible way.
- The selected iterative development model was practical, as the requirements

kept changing. In the beginning of the project, the developers only had a very vague idea on what the search engine should really do, so it would have been difficult to nail the requirements correctly in one try. Also, it would have been difficult for both the team and the client generate a complete picture on what the new system should be and look like, since the understanding on how the corpus could and should be used and what is generally usable and technically possible was only gained through iterating the project tasks.

- Project client found the iterative model practical, too. He greeted that it made it possible to make changes to the plans and functionality all the time - which meant of course that some development rework was required - and yet we could all “remain friends”.

In general, the TamBiC project team learned a lot from this project, from documenting, communication, coding and even some language skills. We discovered that keeping contact with the client is the only way to really understand his needs. TamBiC project was very lucky to get a client that showed immerse interest in the project developments and was were dedicated in guiding the project work and who was always willing to help and review the products.

Team communication is vital to ensure a good ending, if someone is not answering for a long time might be is to scare to say is lost! Especially in international teams, language can be a problem while writing documentation, email or explaining simple tasks. We managed fine and review documents twice before we send.

Analyzing project needs will help you choose better tools to develop it. Sometimes we felt we were in such a hurry to start actual coding that we didn’t take the needed time to analyze the tools. At the end, we found out that there are some tools and methods could have saved us a lot of work. But naturally, that’s something you don’t know at the beginning, and you lack the experience to evaluate what is really needed. Maintaining an organized and clean wiki page saves a lot of time for documentation. We always kept a track of all the changes done so in the end the whole documentation work was almost done “by itself”.

Project statistics

This section describes the project statistics about weekly working hours etc. Most the done project work falls into the “planning” or “coding”. Since we did not have many separate design meetings, but instead, the all regular meetings all included also requirements analysis and UI and code designing planning besides the actual work planning, the “project management” hour figure is rather big. That also explains why so little time was spent specifically only on requirements or design.

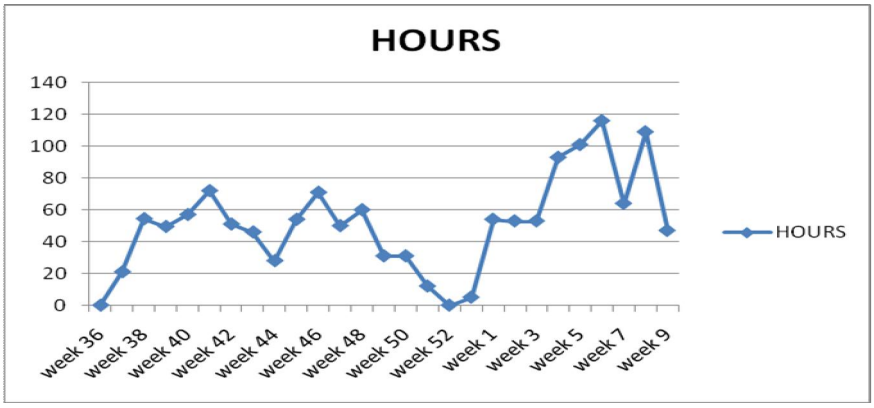
All documentation (i.e. wiki pages) and code was review on iteration basis, which explains the low figures in “reviewing”. As the system was also constantly tested while being developed, also the “testing” figure is rather small; the testing refers mostly to running the planned integration test on the system, near the end of the project.

Team size	Dev. model	Start date	End data	Days	Hours
2+4+1	Iterative	09.09.2009	15.03.2010	189	1383

Table 63: General project information.

Acti-vity	Plan-ning and manag ement	Req. specifi cation.	Design	Code	Integra tion and testing	Re-views	Repair	Study	Other	Total
Hours	355,5	41	83,5	458	74	41	117	154	59	1383
%	26 %	3 %	6 %	33 %	5 %	3 %	9 %	11 %	4 %	100%

Table 64: Effort by activity.

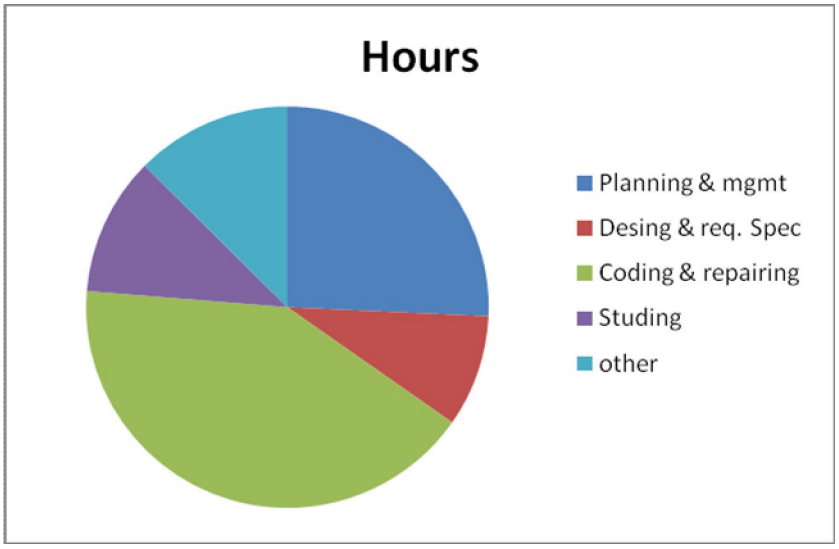


Picture 38: Working hours on weekly basis

	Gina	Harri	Joe	Pekka	Reeta	Simone	**total:**
Planning & mgmt	115	34.5	34	31	119.5	16	**350**
Requirement spec.	17	0	13	0	5	6	**41**
Design	1	50.5	10	15	7	0	**83.5**
Coding	0	81	108	167	51	51	**458**
Integration & testing	18	9	13	0	26	2	**68**
Reviews	36	0	0	0	5	0	**41**
Repairing	11	13	0	15	53	25	**117**
Studying	19	16	33	36	17	29	**150**
Other	17	5	9	1	16	9	**57**
total:	**234**	**209**	**220**	**265**	**299.5**	**138**	**1365.5**

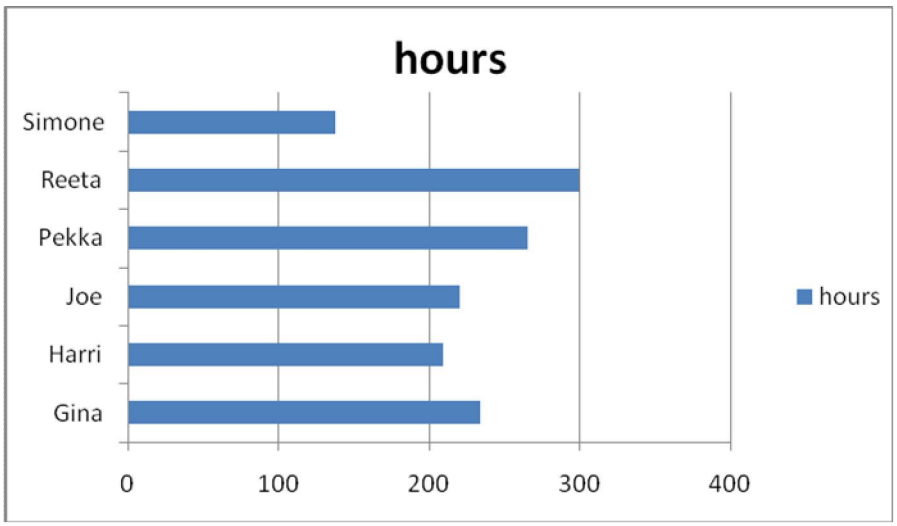
Table 65: Personal participation in activities.

Note that the hours worked by the 2 developers that later quit the project are not included into the figure in Table 3; therefore, the total amount of hours there is smaller than the real one.



Picture 39: Effort by activity

In Table 7, **Design & Req. Spec.** covers categories "Req. specification" and "Design". **Coding and Repairing** covers categories "Code" and "Repair". Category **Other** comprises of "*Integration and testing*", "*Reviews*" and "*Other*".



Picture 40: Total working hours by team member

Documents	Pages	Final Version
Preliminary analysis	9	0.3
Project Plan	38	1.0
Usability analysis	11 +10 more documents related to usability	0.3
Requirements specification	31	1.5
Test plan & report	3 wiki pages	1.0
UI design specification	1 long wiki page	0.8
Implementation description	1 long wiki page	0.5
Usability test report	15 + 9 questionnaires and 3 appendix	1.0

User Manual	9	0.7
Maintenance plan	1 long wiki page	0.6
On-line Help page	1 html page	1.2
On-line About TamBiC page	2 html pages	1.3
Final report	32	1.0
Project's story	14	0.7
Weekly reports	24	*no version control*

Table 66: Project documents.

Language	Java (/JavaScript), PHP, HTML, CSS
LOC (*) does not include re-used code	Total: ~ 5800 LOC o Search engine: ~ 4800 LOC o CSS: 800 o HTML: 460 o Java: 2700 o PHP: 800 o Admin tool: ~ 1050 LOC (*) o HTML: 120 o PHP: 930
SLOC (*) does not include re-used code	TOTAL: ~ 4600 LOC o Search engine: ~3600 SLOC o CSS: 740 o HTML: 370 o Java: 2000 o PHP: 500 o Admin tool: ~ 970 SLOC (*) o HTML: 110 o PHP: 860
Reused code: PHP libraries *(not included in previous LOC/SLOC figures)*	for Admin tool: • PHP: 89 files, 10200 SLOC
Reused and modified	0
Files and classes	Search engine: 21 files • CSS: 3 files • HTML: 3 files • Java: 10 files, 10 classes • PHP: 5 files, 5 scripts Admin tool: 25 files • HTML: 5 files • PHP: 20 files
Functions	Search engine: • Java: 57 • PHP: 22 Admin tool: • PHP: 26
Code revisions	several

Table 67: Code lines.

BirthData

Overview

The goal of project BirthData was to build a system that captures the creation and use history data of virtual objects and presents that information using different modalities if possible. The goal has been to provide more meaning to icons and make them more appealing to the users by reminding them about virtual object's history and creating feeling of object's "life". The application we developed is a file browser with extended features with an emphasis on the concepts. The product will run on Maemo OS in N900 internet tablets with 800x480 resolutions.

Organization and management

The project team consists of project managers taking Software Project management course, team members including one usability team member taking the Project Work course and a graphic designer assigned by Demola.

The project managers are:

- M.Orkun Özen
- Jaspreet Singh

Other team members are:

- Antti Leppänen
- Evgeni Borisov
- Jakub Zlotowski
- Mikhail Kapitonov
- Rauno Tamminen

One of the team members quit the course.
There is also a usability team (U-Team) member assigned for the project:

- Santeri Saarinen (Santeri.Saarinen@uta.fi)

Graphic Designer:

- Vilja Heinonen (Vilja.Heinonen@uta.fi)

Figure 41 – Project BirthData team missing Evgeny, Rauno and Vilja.

Methods and tools

The file browser will run on Maemo OS in Nokia N900 devices with 800x480 resolutions. For programming in Qt 4.5 and OpenGL ES 2.0, the developers used mostly the Qt Creator IDE and Imagination POWERVR SDK.

For documentation, especially when tracking sprint backlogs and weekly reports we benefited from the Google Docs. Google Sites provided the ease of creating and managing a webpage where we uploaded and shared our sketches, documents and files. Especially the integration with GoogleDocs and GoogleCalendar helped with reminding team members about the deadlines, sprint backlogs as well as the upcoming events.

The use of Skype considerably compensated the lack of communication among the remote members. For notifications and matters concerning the whole team, we used the mail group we created.

Project phases and development model

We tried to follow agile software development model, Scrum. We had a total of five planned sprints varying from two weeks to a month depending on the current situation and impeding milestones such as the user study and academic breaks.

Throughout the project we added user story items to the product backlog and suggested a sprint backlog to our clients. At the end of each sprint, the whole team met our client to review the previous iteration and discussed the upcoming sprint's backlog items. We had weekly team meetings on Sundays (!) and client meetings on Tuesdays. This gave us the chance to discuss on urgent matters as a team before the client meeting.

In the first sprint we tried to create the basic application structure and a suitable xml structure that would support any set of metadata we would pick for the icon

visualization.

Starting from the second sprint, we focused more on the information we wanted to present on the icon. In the following sprints, our developers worked on the GUI and the supporting application (what we called the background processes) to collect the data to reflect it to the icon representation. The graphical user interface components were sketched separately by our designers to be presented as ideas during the weekly team meetings. There we picked the best suggestion and refined it before we added the relevant user story to the following sprint.

We didn't have formal documentation apart from the preliminary analysis and the first few versions of the project plan at the beginning. These documents were inspected with the course supervisor as well as the clients. Testing guidelines and UML diagrams for application design were prepared in need.

Actualized risks

The following risks were actualized during the project:

- Project Managers' lack of experience might slow down the progress. Although, the project was never considered behind the schedule and eventually met the client's requirements, with better management the goals could have been met faster and with less effort.
- One team member dropped out of the team after the third sprint.
- Allocated hours exceeded. Some team members greatly exceeded the estimation (circa 200 hours) allocated for each person. Worth noting is that hours were very varyingly divided within the project team. This was due to fact that only part of the team actively participated in the implementation which proved to be the bottleneck.
- English language causes understanding problems: in some cases among the members and also the project managers had difficulties in expressing and understanding with ease.
- Project hours exceeded: despite missing a member, our total hours exceed the estimated count considerably.
- Uneven distribution of hours. Some team members possessed special skills crucial to the project, thus they were assigned to more tasks than some others.

The last risk was not foreseen in the initial risk assessment.

Figure 42. Main screen of the Birthdata-application.

Experiences

We consider this project quite teaching and a good opportunity for collaboration. Although the final application may not satisfy the requirements for a final product, we believe we brought out a decent prototype to give the essence of the proposed idea.

We have had already employed professionals in the business as well as younger students with no real work experience. Therefore, especially for some of us, this opportunity served as a good frame of reference for a real world project. We enjoyed the collaboration within the team and also with our clients with their guidance both in management and concept design.

Working on Maemo development has been quite challenging but it offered the chance to gain experience on this emergent environment. Although they have encountered quite a few problems throughout the project, our developers were motivated for most of the time.

We have been quite a multicultural team so we occasionally had difficulties in working together and tried to overcome the misunderstandings.

We had a member who wouldn't spend much effort on the project and later on decided to quit. This decision was in favor of the team because the remaining members were more motivated to share tasks and put their efforts in an equal manner. Also the managers started to make more accurate estimations using the remaining resources.

However, as managers, we initially had planned to include 4 members in the development team and spare our hours for management. Later on, only two developers could work intensively on the project proving that we could have used our managers as developers as well sharing the workload.

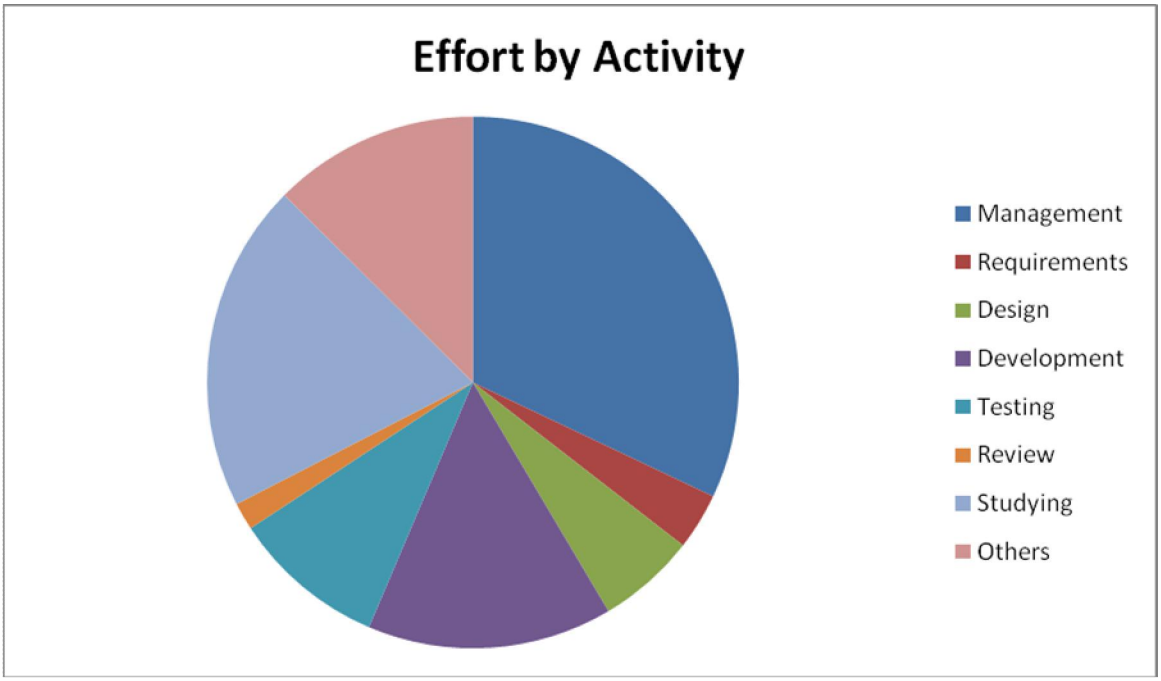
Our usability member and the designers share some of their experience saying it is a must to prepare really well for any field study and make sure the testing and bug fixing is carried out long before the test is to be given a start.

Our developers agree that they could have set the tools such as the version control for collaboration much earlier and actually used it. Because they admit that most of the times they were a bit timid or reluctant to publish unfinished code-work. Also they say they could have worked together more often but given the nature of this project work course it wouldn't have been easy.

Statistics

Team size	Dev. model	Start date	End data	Days	Hours
2+5+2	Scrum	13.09.09	11.03.2010	183	~1650

Table 68: General project information.



Activity	Planning and management	Req. specification.	Design	Code	Integ-ration and testing	Reviews	Study	Other	Total
Hours	535.5	46.5	101.5	240	162	25.5	323.5	170.1	1611.6
%	33.2	2.9	6.3	14.9	10	1.6	20.1	10.6	

Table 69: Group effort by activity.

Number of requirements	Pages	Use-cases	UI screens	Database diagrams	Database tables
81	-	-	-	1	1

Table 70: Requirements and high-level design outcomes.

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

Table 71: Design outcomes.

Document	Pages	Versions
Preliminary analysis	13	3
Project Plan	21	5
Usability analysis	9	3
Product Backlog	1	-
Design Specification	13	1
User interface document	12	1
Test plan	8	2
Test report	1	1
Usability test report	33	5
Final report	14	1
Project's story	7	1
Weekly reports	20	-

Table 72: Documents.

Birthdata(Main Code)

Language	files	blank	comment	code	scale	3rd gen. equiv
C++	67	2193	1565	8130	x 1.51	=12276.30
C/C++ Header	45	823	302	2223	x 1.00	=2223.00
make	5	262	75	1421	x 2.50	=3552.50
XML	1	3	0	54	x 1.90	=102.60
Bourne Shell	3	3	0	12	x 3.81	=45.72
SUM:	121	3284	1942	11840	x 1.54	=18200.12

birthdata-process

Language	files	blank	comment	code	scale	3rd gen. equiv
C++	7	304	99	1269	x 1.51	=1916.19
C/C++ Header	4	71	13	216	x 1.00	=216.00
make	1	43	15	211	x 2.50	=527.50
SUM:	12	418	127	1696	x 1.57	=2659.69

Table 73: Codelines.

Rich context information sharing in Facebook

Overview

This project implemented a Facebook application to display context information obtained from a mobile phone. The company partner, Nokia Research Center, provided us a software package that detects the user's environment (e.g. car, restaurant, street) and activity (e.g. idle, running, walking). The mobile application transmits this information to Facebook, and a Facebook application displays the information in an entertaining and interesting way.

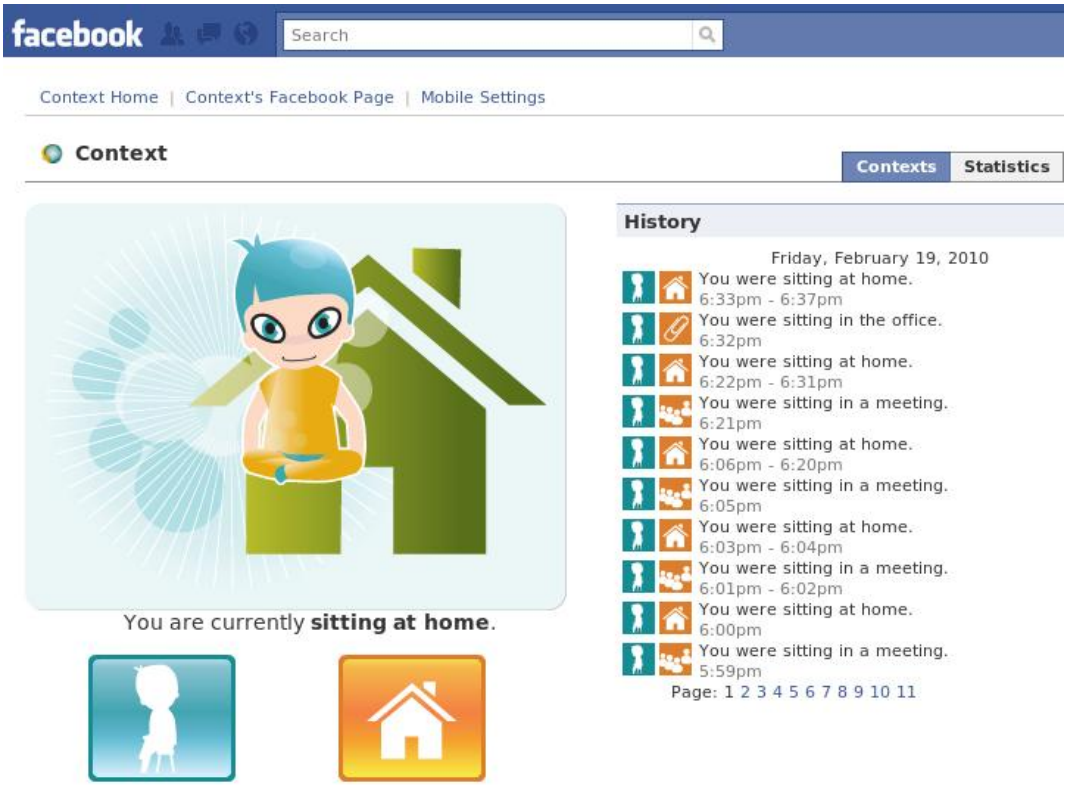


Figure 43: Facebook application.

Organization and management:

Member	**Role**
Stanislav Radomskiy	Manager (Mediator team)
Gururaj Mahajan	Manager
Jean Fairlie	Manager (Mediator team)
Mervi Ollikainen	Developer (Facebook team)
Jan Stepien	Developer (Server team)

Andrew Knight	Developer (Facebook team)
Marisela Gutierrez	Developer (Usability team)
Mikko Sauna-aho	Developer (Mediator Team)
Ville Pylkki	Developer (Usability Team)
Vilja Heinonen	Graphic Designer
Juhani Vainio	Usability team leader

Figure 1: Context Team (Jan, Vilja, Guru not present) .

The workload was distributed among 3 self-organizing development teams, which focused their work on the 3 main modules of the application. Each team had a great amount of freedom as well as responsibility, and was responsible for its own development tools selection, requirements elicitation, implementation and testing.

Each team had a skilled and motivated leader and between 1 and 3 developers. Each team leader was responsible for the work of his team, and it was his duty to identify, organize and distribute tasks among his subordinates. In addition, team leaders were responsible of communicating with the other leaders to define the interfaces that integrated each team's modules into one functional application.

Since team's co-location was unfeasible in this context, and documentation ceremony was relatively light, communication problems were expected. Therefore we constructed an environment where we had strong communication channels despite our almost totally remote work dynamics and uneven schedules.

1. A 24/7 Skype based logged group chat was used for everyday communications.
2. A mailing list was set up (context@cs.uta.fi)
3. Weekly (whole) team physical meetings were held to assess progress, as well as discuss and resolve issues.
4. Heavy use of specialized collaborative tools (Git, Google Docs, Wiki, Trac, Hourcount)

Methods and tools:

Local Development Environment and Tools		Usefulness
Languages	Qt for S60 Ruby Standard SQL PHP5 FBML (XHTML/CSS), JavaScript, ActionScript/Flash	
Editors/IDEs	Notepad++ Vim Eclipse, FlashDevelop, Illustrator, Photoshop	
OS(where developers work)	Windows XP Windows Vista GNU/Linux	
Production, Test Environments and Tools		
OS (where applications run)	Symbian OS S60 3rd & 5th Edition FP1 Debian GNU/Linux 5.0	
Devices	Nokia N95	
Emulators	S60 3rd Edition SDK for Symbian OS Emulator	
Databases	MySQL (production) PostgreSQL (testing) SQLite 3	
Web Server	lighttpd Sinatra	
Mobile Browsers	S60 OSS Browser	
Browsers	Firefox 3.x, Internet Explorer, Safari, Internet Explorer, Google Chrome	
Management and Other Tools		
Revision Control	Git, Gitorious	Interesting but maybe too complicated for our simple needs
Bug Tracking and Wiki	Trac	Very useful
Documentation	OpenOffice.org, Google Docs	Google Docs was very useful
Time Management	Hourcount	Very helpful
Communication	Skype, mailing list	Very helpful

Project phases and development model

We used a specially tailored development model that resembles agile methodologies in their incremental and iterative approach, fixed schedule and flexible scope, but which is similar as well, to more traditional models (i.e. hierarchical structures, less democratic environments, less dependency on team members pro-activity and co-location).

The project was broken down into 7 iterations, each between 2 and 3 weeks long. Features were frozen at the end of the sixth iteration. And the last iteration was used to fix bugs, finish up the product and prepare the final presentation.

Each iteration had roughly, the following phases:

1. Requirements definition (based on the previous iteration's feedback, or initial discussions with the client) and work distribution.

2. Coding, test design and test execution.

3. Client meeting and feedback group discussion.

We didn't have any formal documentation other than the "Preliminary Analysis" and "Project Plan" at the beginning of the project. All relevant information such as requirements, design , testing, and client reviews were documented incrementally (if at all) and put in our project's wiki page.
The "formal" documentation was created "a posteriori" during the 7th and last iteration of the project.

Foreseen and unforeseen risks met

Out of 12 foreseen risks at the beginning of the project it seems only 3 materialized:

(1) Project managers cannot split the tasks between each other evenly.

This problem did affect us, since it was just one of the three managers who ended up doing most of the managing tasks, and a second manager helping the first one in some few tasks. The management workload was incorrectly perceived as small and it was deemed impractical to split most of the work, to such extent that 2 of the managers ended up taking coding tasks and a third manager losing interest and almost not participating in the project at all. It is now clear that some tasks that could and should have been assigned to one of the other managers were never assigned and in some cases even addressed properly (e.g. testing, documentation).

(2) Lack of programming skills of some of team members might slow down the progress, and

(3) Team member is too busy with other work or study to accomplish the task

Thanks to the relatively simple nature of the product, and the efficiency of the more active team members, progress was not actually affected, but it did happen that some of the project members couldn't really fit in the process and contribute in a meaningful way, whether they didn't have the necessary skills, time, motivation or some combination of those, it seems clear that most of the work was carried out by as little as 50% of the team.

As for the unforeseen risks, we failed at identifying the task of creating a stronger communication channel between the usability team (UT) and the development team as a possible issue. This problem made itself clear at the end of the project when we realized that some of the results of the UT's analyses were not taken into consideration in the final product.

Experiences

The project was an interesting experience for most of us, we had the opportunity to work on a real project within a controlled environment, without the risks and necessary time commitments involved in real life projects.

We all have learned something from all these months working together, whether it is a new technology like Qt or the Facebook API, a new version control tool like Git, or just new insights into how to tailor an efficient software development workflow.

Here are the comments of one of our team members:

"I had never programmed and shared code in a team before, but it was a good experience. It turned out that making an application had a lot of different sides to it from planning to testing, and new challenges kept popping up in each iteration. Also the project taught that applications are never so ready that they don't need updates or improvement in the future. There are many things to consider whilst making a program, for example just translation can cause a lot of issues."

In general we think the project has been successful in meeting our client's expectations, and we are happy with the results, however there are a few things that we know we could have done better.

Managers feel they should have addressed some important tasks from the beginning. For instance, the testing process lacked concrete planning and was many times carried out, especially with the GUI, in an improvised way. In a similar manner, some tasks related to the project documentation were neglected until the last iteration, creating unnecessary stress on the team at the end of the project. Members of the team involved in the usability analysis and user studies feel they should have tried to integrate a little better in the whole development process, making the results of their analyses have more influence in the final product.

All in all, many members feel they could have been more active and taken more responsibilities. Hopefully in our next project we all will.

Statistics

This section describes various statistics related to our project.

Team size	Dev. model	Start date	End date	Days	Hours
3+6+2	iterative	28.09.2009	11.3.2010	85 working days	1303

Table 74: General project information.

Activity	PM	Req.Spec	Design	Code	I&T	Reviews	Repairs	Study	Other	Total
Hours	398.5	14.5	20	370	55	52	20.5	71.5	239	**1303**
%	30.5	1.1	1.5	28.3	4.4	3.9	1.5	5.4	18.3	

Table 75: Group effort by activity.

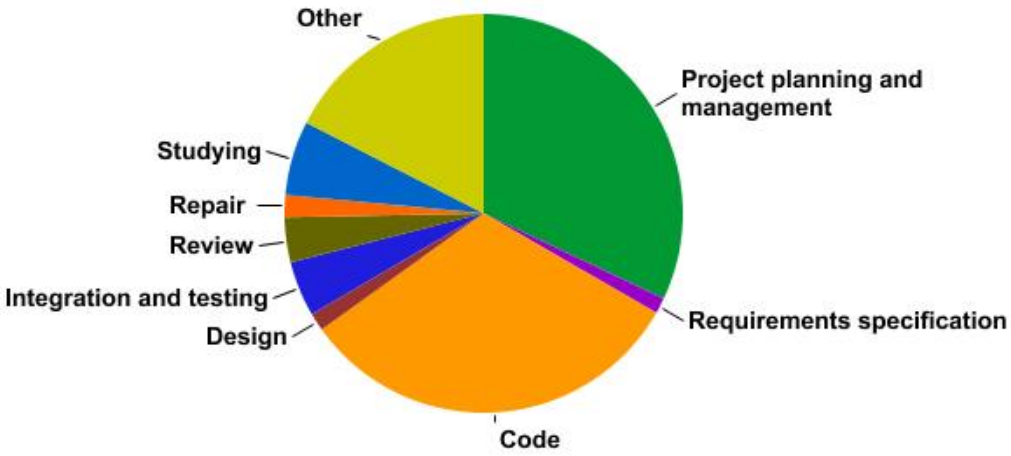


Figure 44: Effort by Activity

Number of requirements	Pages	Use-cases	UI screens	Database diagrams	Database tables
45	45	0	0	1	1

Table 76: Requirements and high-level design outcomes.

Pages	Overview diagrams	Class diagrams	Sequence diagrams	State diagrams	Other diagrams
5	1	0	0	0	3

Table 77: Design outcomes.

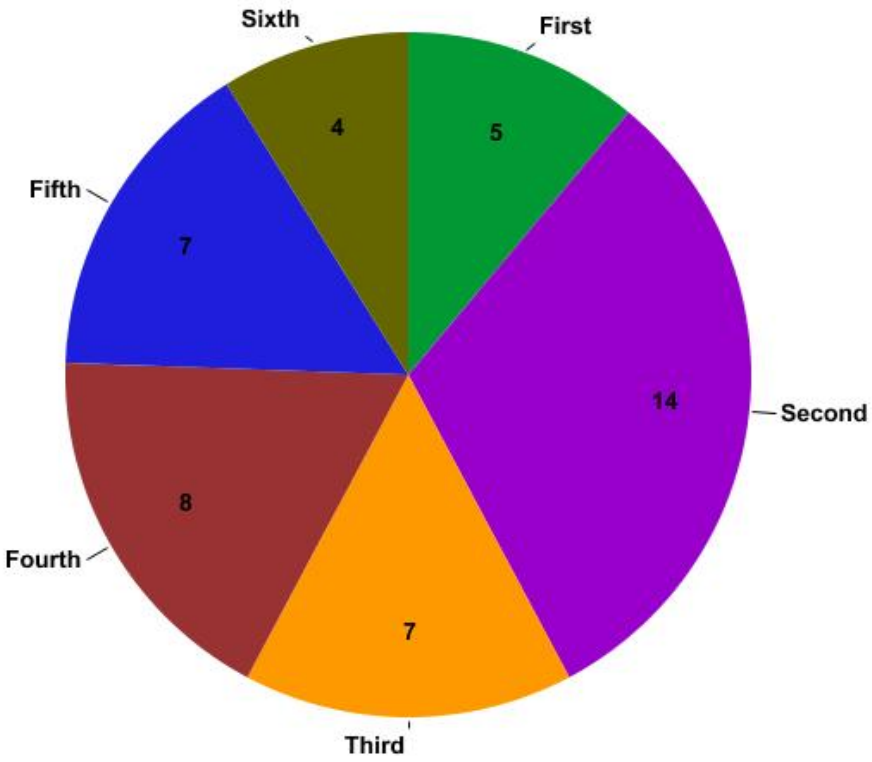


Figure 45: Requirements per Iteration.

Document	Pages	Versions
Preliminary analysis	6	4
Project Plan	17	7
Usability analysis	10	3
Requirements specification	N/A*	N/A
Design Document	5	1
Test Cases	8	2
User Study Plan	12	6
User Study Report	8	4
Final report	11	2
Project's story	8	2
Weekly reports	17	1

Table 78: Documents (* In web format, no printable version).

Language	C++, PHP, Actionscript, RUBY
SLOC	6483
Classes	30
Functions	245
Code revisions	201

Table 79: Codelines.

Collective cognitive state of people

Overview

The project name is Collective Cognitive State of People. It is a project provided by Nokia which aims at gathering anonymous data from people regarding their state. At the moment the information gathered is limited to people's mood as well as some additional personal information (age and sex). No names or other information that could identify a person is gathered.

The implementation of the idea is done through mobile devices. At the same time there is a webpage that visualizes the information gathered. Some features of the project implementation include:

- Information gathered: gander and age information, user's mood, picture and an optional comment.
- The webpage features a map of user moods and some visualizations of the statistics gathered (graphs).

Organisation and management

Project members:

- Tausif Baber (manager)
- Rajib Paudyal (manager)
- Jefim Borissov (manager)
- Saravanan Dhanabal (developer, mobile)
- Avishek Barua (developer, mobile)
- Mikko Hissa (developer, web, server management)
- Antero Mäenpää (developer, web)
- Joonas Jokiniemi (user interface and usability)
- Szymon Piskula (developer, web; exchange student, left the project in December 2009)
- One developer quit the project in the beginning.

Methods and tools

There were several different tools used in the project. Their description and usefulness

is described below:

- Subversion & Tortoise SVN
    - Subversion is a source versioning tool that our project used to cooperate and work on the project collaboratively. It is widely used throughout the world.
    - As an additional tool we used Tortoise SVN which is a client application to work with Subversion repositories.
        - The combination of those two tools was extremely helpful and contributed a lot to the collaboration capabilities of the project members. Version tracking and syncronisation is very important in any project.
- MediaWiki
    - MediaWiki is a wiki that our project utilizes to keep track of the progress as well as write documents related to the project (project analysis, test plans etc.).
        - Our experience with MediaWiki is very positive - every piece of information about the project and its progress can now be found in the Wiki and the process of adding new information is very easy. All in all, this was a very good tool, which we used a lot. For programmers it was probably tool number two in their daily routine and for managers - number one.
- Apache Tomcat
    - The server-side solution used to power the web part of the solution. This is widely known software used all around the world and it did very well since our web application is written on Java.
- Nokia SDK
    - The SDK (Software Development Kit) for Nokia mobile phones is a must for phone development. It was a good experience.
- Carbide C++ 2.0

    - Carbide is an IDE for mobile development used in our project and it is based on Eclipse. Therefore it is a very stable product without any noticeable flaws in it. We had a good time using it to develop the mobile part of the project.

- Nokia 5800 phones
    - Used for testing the project.
        - It was nice having the target devices right away, yet the most use for this was closer towards the end of the project, when we could actually see that the solution works and if it has any considerable flaws. In the beginning of the project, on the other hand, programmers were heavy users of those phones as mobile development requires testing on a regular basis on the target device due to the amount of different devices (and specific differences between them).

Project phases and development model

In this project we used Scrum as our development model. It is an agile methodology, which allowed the whole group to easily get started with development and get solid tangible results in a short period of time. Every Sprint review was something to wait for and see how it went. We had the opportunity to assess our skills and capabilities thus making the next iteration more exact and complete. In the end we could quite exactly define the requirements and outcomes of each Sprint.

There was a considerable amount of time spent in the beginning of the project for setup and getting the team together but in the end we started doing the project. And the first thing we did is we met the client to talk about the project.

When the first official sprint started we already had an image of the project and some tools online (wiki). The whole process was interesting and new to some of the project members. Some difficulties were encountered, including some lack of skills and the fact that a lot of members in the project are working at full-time jobs as well as studying at the same time, so we had a hard time setting up our meeting schedule. But in the end we got to real work and coding and started gradually implementing the features and requirements we got from the client.

During sprints we did not have anything really exciting – just regular planned work and reviews in between. Everything went quite calmly and without major problems. Two factors were there to disrupt our work though: in the beginning of the project a member of our project left the course. This was not a big problem since we did not yet start the actual work, so we were able to easily adjust to the new situation. The second problem was connected to our other project member – Szymon. He is an exchange student from Poland and he had to leave the project in December. This was also a very

easy problem as we knew this in the beginning of the project, so we were prepared.

In the end of the project most of the team's effort was focused on fixing bugs and improving the existing functionality. The project managers were also quite busy as they were preparing the documentation for the final project CD.

As a physical result of the project we created an application for mood gathering.

Project phases

Phase	Start	End	Comment	In charge
Starting the project	14.09.2009	03.12.10	Documentation	Managers
Sprint 0	12.10.2009	02.11.2009		All members
Sprint 1	02.11.2009	23.11.2009		All members
Sprint 2	23.11.2009	14.12.2009		All members
Sprint 3	14.12.2009	11.1.2010		All members
Sprint 4	11.1.2010	01.2.2010		All members
Sprint 5	01.2.2010	15.2.2010		All members
Sprint 6	15.2.2010	01.3.2010		All members
Finishing the project	15.2.2010	12.3.2010	Documentation	Managers

Experiences

The overall experience from the project was very good as we got to work with a real project from Nokia. It involved a lot of planning and thinking for the managers and developers as well as maintaining the links between all parties involved (communication within the team, collaboration tools, client meeting, reviews etc.). So in the end all of us benefited in some way or another - developers had time to learn new technologies and play around with mobile phones and mobile development and managers got some image of how it feels to try to be on top of something big. Organizing meetings, keeping track of project progress and statistics, solving issues and in some way even trying to motivate other people.

To summarize, we had a great time doing this project and it was very nice to explore this area and to work with a real client. At the same time we met new interesting people and got to know how a project work from inside out and were able to apply our knowledge in practice. We would like to thank both Demola and Nokia and their representatives – Ville Kairamo and Jari Kangas respectively for giving us this opportunity.

Statistics

Team size	Dev. model	Start date	End date	Weeks	Hours
3+5+1	Scrum	12.10.09	12.03.10	25	1040

Table 80: General project information.

Activity	PP&Man	Req-s	Design	Code	Integ-ration	Rev-iews	Re-pair	Study	Other	Total
Hours	265,5	34	10	182,5	58	66	57,5	144,5	106,5	929
%	28,7	3,7	1,1	19,8	6,3	7,1	6,2	15,6	11,5	100
Usability	29,5	0	14,5	0	0	11		6,5	49,5	111
Total	300	34	24,5	182,5	58	77	57,5	151	155,5	1040

Table 81: Group effort by activity.

Number of requirements	UI screens
18	6

Table 82: Requirements and high-level design outcomes.

Document	**Pages**
Preliminary Analysis	8
Project Plan	18
Usability analysis	11
Retrospective on User Experience	19
First User Study	5
Second User Study	9
Requirements specification	5
Test plan	8
Final report	27
Project's story	8
Weekly reports	12

Table 83: Documents.

Used tools	QT for Symbian S60, Symbian S60 SDK 5th Edition, Carbide C++ 2.0, Symbian S60 / QT mobile extension preview 2, Apache Tomcat, MySQL, ExtJS
Language	C++, Java, JavaScript, HTML, CSS
LOC* (Web)	~500
SLOC* (Web)	~300
LOC* (Mobile)	~1200
SLOC* (Mobile)	~1030

Table 84: Codelines.

* LOC - Lines of code

* SLOC - Source lines of code

Tracker

Overview

Tracker is a simple issue tracker inspired by Pivotal Tracker, which is a free to use issue tracker usable on the Internet (www.pivotaltracker.com). Original plan was to create similar Scrum project manager to JIRA issue tracking software created by Atlassian, but due to difficulties faced during the project it was decided that the final product could not be a JIRA plugin. Tracker is specifically designed to be used with Scrum software development model, so it features all the relevant concepts that are present in Scrum such as product and sprint backlogs, sprint length and velocity. The objective was to create an easy to use and lightweight program with an emphasis on rapid creation of issues/stories. The program can be deployed to any server with a web server, MySQL and PHP installed and is used with a web browser.

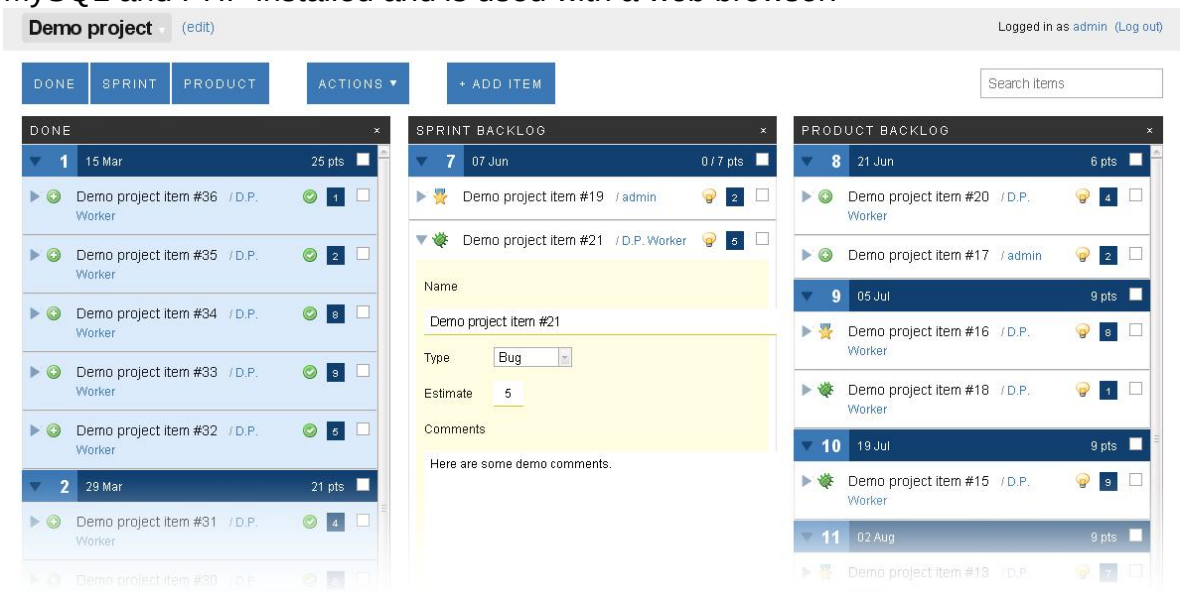


Figure 46. Screenshot of the graphical user interface.

Organisation and management

Originally project had a company client, but due to some problemsAfter the original plan had to be scrapped. Timo Poranen from Department of Computer Science, University of Tampere took the client's role.
The team itself consisted of following personnel:

- Timi Antere - project manager
- Juha Mäenalusta - project manager
- Ville Autio - documentation, testing, design, coding
- Antti Kantola - coding, design, testing, documentation
- Tommi Koskenoja - UI design, coding, design, testing, documentation
- Sanna Kangas (Usability team) - UI design

Team members dropped out of the project. Project was managed mostly through e-mail, IRC and face to face meetings. Project managers handled the big picture and the project members took tasks that were related to their strengths. Most of the heavy coding work was related to the user interface which was done by Tommi. Antti did most of the server administration and used his expertise with PHP and CodeIgniter to get the server backend working. After the initial plan was scrapped Ville mostly contributed by working on the project documentation and testing the software.

Methods and tools

The programming language was supposed to be Java, but after the client changed, this too was changed. The software was implemented using JavaScript and PHP languages. The development was aided by the use of jQuery and CodeIgniter frameworks, jQuery for JavaScript and CodeIgniter for PHP. CodeIgniter provided the groundwork for MVC (Model-View-Controller) software pattern, input validation, database integration and user management, among other things. JQuery was heavily used in the implementation of the user interface, which consisted of normal HTML code and ready UI components available in jQuery. The user interface is based on the AJAX (asynchronous JavaScript and XML) principle, which is easily implemented through the use of jQuery.

Google Docs was used for collaboration on the project documents and tracking working hours. Main means of communication was e-mail which was supplemented with the use of IRC. Doodle was a great help in arranging meeting times. Generally all these tools worked as expected. PivotalTracker was also used, mainly because the UI needed to be similar. It was not a valuable tool for the team.

Project phases and development model

Development model used in the project was Scrum, which wasn't followed to the letter due to the nature of the project. Instead of daily meetings weekly IRC meetings held in the beginning of the project and sprints were informally reviewed every two weeks. There were four formal reviews held during the project which were very useful despite the lack of goods to show. Due to the technical difficulties real, structured development never got started and the final product was largely rushed together after having the web user interface nearly finished for a few months.

The project plan was radically changed two times during the project which changed the direction of the project. Before the original plan was scrapped we had a good idea of what needed to be done and we had a nice and working user interface prototype available. After facing major issues with JIRA and original project deadline near, a plan B was created which unfortunately also involved JIRA. Faced with the same issues, the project stalled again and after a crisis meeting it was decided that we would do a standalone implementation instead of scrapping the whole project. With the user interface in good shape and a couple of team members with good PHP skills a server backend was then quickly created. The software was deemed good enough in May after two (relatively) intensive coding sessions.

The original plan mostly failed because of major difficulties with JIRA. Atlassian was just about to release a major new version and they had also just changed their plugin

system to a new version, which meant that their tools were in the first beta version when our project got started and the documentation available was mostly out of date. For the most part the plugin tools didn't work as they were supposed to, if they worked at all. Due to lack of communication from the client and bad documentation the team really didn't know how to go about implementing the software. The team also suffered from bad communication and mismanagement of personnel which combined with the technical issues lead to motivation issues, further compounding the other problems. For the most part the technical problems weren't anticipated and while the motivation issues of this magnitude, which were largely a result of the technical problems, weren't probably expected it probably didn't come as a much of a surprise that they emerged.

Experiences

The project really did not go as expected and hence it was mostly a bad, if teaching, experience for the people involved. Facing multiple technical difficulties was very demoralizing as a lot of time and energy was spent on trying to solve them instead of getting to write actual code. The client was not very supportive and in hindsight the original task probably should have been turned down as too difficult or time consuming to implement in the context of this course. Fortunately there were positive experiences too, the lecturer was very flexible and supportive when the team was struggling and the few times we got together to actually work together on the actual code were quite productive and done in good spirit.

So in hindsight it is easy to say that there was room for a lot more collaboration and effort and we probably should have created a standalone prototype first before starting to struggle with JIRA. The team had a quite few people with relevant work experience and the technical competence was relatively high, but just not on the technologies that the initial client wanted to be used, so without the aforementioned problems the result might have been quite different.

Statistics

Team size	Dev. model	Start date	End date	Days	Hours
2+3+1	Scrum	2.9.2009	10.6.2010	282	533

Table 85: General project information.

Activity	Plan. and man.	Req. spec.	Design	Code	Integ-ration and testing	Reviews	Re-pair	Stu-dy	Ot-her	Total
Hours	224	8	91	87.5	2	8.5	0	91.25	41	**533.25**
%	40	1%	16%	16%	0.5%	2%	0%	16%	7%	

Table 86: Group effort by activity.

Number of requirements	Pages	UI screens	Database diagrams	Database tables
17	1	3	1	6

Table 87: Requirements and high-level design outcomes.

Document	Pages	Versions
Preliminary analysis	8	1
Project Plan	13	2
Usability analysis	8	1
User interface document	8	1
Test plan	8	1
Final report	12	1
Project's story	5	1
Weekly reports	30	27

Table 88: Documents.

Language	files	Blank	comment	code
PHP	20	197	103	1464
Javascript	1	62	36	633
CSS	3	34	5	609
SQL	1	15	3	102
SUM:	25	308	147	2808

Table 89: Codelines.