

Verso-sovellusprojekti

Projektiraportti

Tero Hänninen

Juho Nieminen

Marko Peltola

Heikki Salo

Versio 0.10.0

Julkinen

10.12.2010

Jyväskylän yliopisto

Tietotekniikan laitos

Jyväskylä

Hyväksyjä	Päivämäärä	Allekirjoitus	Nimenselvennys
Projektipäällikkö	__.__.2010		
Tilaja	__.__.2010		
Ohjaaja	__.__.2010		

Tietoa dokumentista

Tekijät:

• Tero Hänninen (TH)	tejohann@jyu.fi	0400-240468
• Juho Nieminen (JN)	juho.nieminen@jyu.fi	050-3831825
• Marko Peltola (MP)	marko.peltola@jyu.fi	041-4498622
• Heikki Salo (HS)	heikki.ao.salo@iki.fi	050-3397894

Dokumentin nimi: Verso-projekti, Projektiraportti

Sivumäärä: 67

Tiivistelmä: Verso-projekti toteutti Jyväskylän yliopiston tietotekniikan laitokselle prototyypin lähdekoodien julkistamisjärjestelmästä nimeltä YouSource. Dokumentti kuvaa projektin taustaa, resursseja ja läpivientiä. Dokumentissa tarkastellaan myös suunniteltujen tavoitteiden, käytänteiden, aikataulujen ja riskien toteutumista.

Avainsanat: aikataulu, Git, Gitorious, ketterä ohjelmistokehitys, kokemukset, käytänteet, laadunvarmistus, oppiminen, projektin läpivienti, projektiorganisaatio, projektiraportti, resurssit, riskit, sovellusprojekti, tavoitteiden toteutuminen, tehtäväjako, tehtävät, testaus, Trac, työmäärät, työnjako, versiohallinta.

Muutoshistoria

Versio	Päivämäärä	Muutokset	Tekijät
0.0.1	21.4.2010	Ensimmäiseen luonnokseen dokumentista kirjoitettiin taustaa ja tavoitteiden toteutumista kuvaavat luvut.	HS
0.0.2	23.4.2010	Kuvattu oppimistavoitteiden ja tulosten toteutumista.	HS
0.0.3	29.4.2010	Korjattu termejä.	HS
0.0.4	4.5.2010	Kuvattu organisaatio ja resurssit sekä tehtävien jakautumista.	HS
0.1.0	5.5.2010	Korjattu lähdeviitteet.	HS
0.1.1	6.5.2010	Lisätty kuvia tekijöiden ajankäytöstä.	HS
0.1.2	19.5.2010	Kirjoitettu alustavat luvut käytänteistä, Trac-käytänteistä ja prosessimallista. Tehty pieniä Santasen ehdottamia korjauksia.	HS
0.1.3	24.5.2010	Muokattu lukujen järjestystä luontevammaksi. Lisätty luku yhteistyöstä Gitorious-yhteisön kanssa. Tehty pieniä Santasen ehdottamia korjauksia.	HS
0.2.0	25.5.2010	Lisätty laadunvarmistusta käsittelevä luku. Parannettu taustaa kuvaavaa lukua ja muokattu rakennetta luontevammaksi.	HS
0.2.1	27.5.2010	Lisätty versiohallinnan käyttöä kuvaava luku. Täydennetty Trac-käytänteitä kuvaavaa lukua.	HS
0.3.0	28.5.2010	Muokattu rakennetta. Tehty pieniä Santasen ehdottamia korjauksia. Lisätty jäsenten kokemuksia kuvaava luku.	HS
0.3.1	28.5.2010	Tehty pieniä korjauksia.	HS
0.3.2	31.5.2010	Tehty pieniä Santasen ehdottamia sisällöllisiä korjauksia kaikkiin lukuihin ja muokattu dokumentin rakennetta niiden perusteella.	HS
0.3.3	31.5.2010	Päivitetty avainsanoja ja termejä sekä poistettu versiohistoriasta ilmaisu "Lähetetty tarkistettavaksi". Lisätty Tero Hännisen ja Juho Niemisen kokemukset projektista.	HS

Versio	Päivämäärä	Muutokset	Tekijät
0.4.0	6.6.2010	Kirjoitettu riskien toteutumista kuvaava luku. Täydennetty tavoitteita kuvaavaa lukua.	HS
0.4.1	14.6.2010	Täydennetty vaatimusten toteutumista kuvaavaa lukua. Korjattu pieniä virheitä ja tehty pieniä parannuksia Santasen palautteen perusteella.	HS
0.4.2	21.6.2010	Lisätty uudet Gantt-kuvat projektin vaiheista sekä päivitetty termejä, aikataulua ja prosessimalia kuvaavia lukuja. Laajennettu Teron kokemuksia projektista.	HS
0.5.0	22.6.2010	Tehty pieniä korjauksia.	HS
0.5.1	9.8.2010	Lisätty Heikki Salon kokemukset projektista. Kirjoitettu aloittamattomat luvut ajankäytön analysointia lukuunottamatta.	HS
0.5.2	10.8.2010	Lisätty Marko Peltolan kokemukset projektista.	HS
0.5.3	11.8.2010	Päivitetty Heikki Salon kokemuksia projektista.	HS
0.6.0	17.8.2010	Päivitetty riskien kuvausta, projektin aikakaavioita ja jäsenten henkilökohtaisia ajankäyttökuvauksia.	HS
0.6.1	22.9.2010	Tehty pieniä parannuksia.	HS
0.6.2	4.10.2010	Päivitetty kansiolistaa, tarkennettu oppimistavoitteita, tehty pieniä rakenteellisiä muutoksia sekä lisätty suunnitellun ja toteuman vertailua useisiin alalukuihin.	HS
0.6.3	7.10.2010	Lisätty useisiin lukuihin suunnitelman ja toteuman analysointia.	HS
0.6.4	13.10.2010	Lisätty graafit ja analyysit jäsenten henkilökohtaisesta ajankäytöstä, analysoitu projektin vaiheita sekä tehty pieniä ulkoasuparannuksia.	HS
0.7.0	25.10.2010	Päivitetty suunniteltua ja toteutunutta ajankäyttöä kuvaava Gantt-kaavio, kirjoitettu projektin yhteenveto sekä täydennetty lähdeviitteiden tietoja.	HS
0.7.1	10.11.2010	Tehty Santasen esittämiä esitystapakorjauksia lukuihin 1-7.	HS

Versio	Päivämäärä	Muutokset	Tekijät
0.8.0	11.11.2010	Santasen esittämiä esitystapakorjauksia lukuihin 8-14.	HS
0.8.1	26.11.2010	Päivitetty kaikki projektin ajankäytön kaaviot. Tehty ulkoasuparannuksia lähdeluetteloon sekä lukuihin.	HS
0.8.2	27.11.2010	Tehty ulkoasu- ja esitystapakorjauksia lukuihin 1-6.	HS
0.9.0	29.11.2010	Tehty ulkoasu- ja esitystapakorjauksia lukuihin 7-14. Täydennetty ajankäytön analyysijä.	HS
0.9.1	9.12.2010	Tehty sovitut kieliasuparannukset, päivitetty tuntikaaviot ja korjattu vastaavasti tuntien analyysit.	HS
0.10.0	10.12.2010	Lähetetty projektiorganisaatiolle hyväksyttäväksi.	HS

Tietoa projektista

Verso-projekti toteutti Jyväskylän yliopiston tietotekniikan laitokselle prototyypin lähdekoodien julkistamisjärjestelmästä.

Tekijät:

- | | | |
|----------------------|-----------------------|-------------|
| • Tero Hänninen (TH) | tejohann@jyu.fi | 0400-240468 |
| • Juho Nieminen (JN) | juho.nieminen@jyu.fi | 050-3831825 |
| • Marko Peltola (MP) | marko.peltola@jyu.fi | 041-4498622 |
| • Heikki Salo (HS) | heikki.ao.salo@iki.fi | 050-3397894 |

Tilaaaja:

- | | | |
|--------------------|---------------------------|-------------|
| • Paavo Nieminen | paavo.j.nieminen@jyu.fi | 040-5768507 |
| • Tapani Tarvainen | tt@it.jyu.fi | 050-3130446 |
| • Ville Tirronen | ville.e.t.tirronen@jyu.fi | 014-2604987 |
| • Tero Tuovinen | tero.tuovinen@jyu.fi | 050-4413685 |

Ohjaajat:

- | | | |
|--------------------------|---------------------|-------------|
| • Antti-Juhani Kaijanaho | antkaij@jyu.fi | 014-2602766 |
| • Jukka-Pekka Santanen | santanen@mit.jyu.fi | 014-2602756 |

Yhteystiedot:

- | | |
|----------------------|---|
| • Sähköpostilistat | verso@korppi.jyu.fi ja
yousource-users.group@korppi.jyu.fi. |
| • Sähköpostiarkistot | https://korppi.jyu.fi/kotka/servlet/list-archive/verso/ ja
https://korppi.jyu.fi/kotka/servlet/list-archive/yousource-users.group/ . |

Sisältö

1	Johdanto	1
2	Termejä	2
2.1	Projektin läpivientiin liittyviä termejä	2
2.2	Sovellukseen liittyviä termejä	2
2.3	Git-versiohallintaohjelmiston termejä	3
3	Tavoitteiden toteutuminen	4
3.1	Taustaa	4
3.2	Sovellukselle asetetut tavoitteet	5
3.3	Tavoitteiden muutokset projektin kuluessa	6
3.4	Sovelluksen jatkokehitys	7
3.5	Projektin tulokset	8
3.6	Oppimistavoitteet	9
3.7	Tekniikoista ja ohjelmistoista opittua	10
4	Organisaatio ja resurssit	11
4.1	Projektioorganisaatio	11
4.2	Projektin tilat ja laitteet	12
4.3	Ohjelmointi- ja dokumentointityökalut	12
4.4	Luennot ja perehdytykset	13
5	Yhteistyö Gitorious-yhteisön kanssa	15
5.1	Ongelmien ratkaiseminen	15
5.2	Tulosten julkistaminen yhteisölle	15
6	Käytänteet	17
6.1	Tiedotus	17
6.2	Palaverit	18
6.3	Dokumentoinnin käytänteet	19
6.4	Julkistettujen tulosten versionumerointi	19
6.5	Luovutettujen tulosten hakemistorakenne ja tiedostojen nimeäminen	20
6.6	Tulosten koostaminen ja toimittaminen	21
7	Sovellukseen liittyvä laadunvarmistus	22
7.1	Lähdekoodin käytänteet	22

7.2	Pariohjelmointi	23
7.3	Integraatiotestaukset	23
7.4	Automaattiset poikkeustiedotteet	24
7.5	Esikäyttäjien hyödyntäminen	24
7.6	Käytettävyyispäivä	25
7.7	Käytettävyytestaukset	26
7.8	Koodikatselmoinnit	26
7.9	Järjestelmätestaukset	26
8	Versiohallinnan käyttö	28
8.1	Suunniteltu haarojenkäyttöstrategia	28
8.2	Havaitut puutteet versiohallinnan käytössä	29
8.3	Soveltuvampi haarojenkäyttöstrategia	30
8.4	Tietovaraston rakenteen jatkokehitys	31
9	Trac-käytännöt	32
9.1	Käyttöoikeudet	32
9.2	Tikettien muutoksista tiedottaminen	32
9.3	Tiketteihin kirjatut tiedot	33
9.4	Tikettityypit	33
9.5	Prioriteetit	34
10	Tehtävien työmäärät ja työnjako	36
10.1	Tehtävien jakautuminen	36
10.2	Ryhmän ajankäyttö tehtäväkokonaisuuksittain	39
10.3	Juho Niemisen ajankäyttö tehtäväkokonaisuuksittain	41
10.4	Tero Hännisen ajankäyttö tehtäväkokonaisuuksittain	42
10.5	Marko Peltolan ajankäyttö tehtäväkokonaisuuksittain	43
10.6	Heikki Salon ajankäyttö tehtäväkokonaisuuksittain	44
11	Prosessimalli ja projektin aikataulu	45
11.1	Prosessimalli	45
11.2	Aikataulu	47
11.3	Ryhmän ajankäyttö viikoittain	50
11.4	Tero Hännisen ajankäyttö viikoittain	51
11.5	Juho Niemisen ajankäyttö viikoittain	52
11.6	Marko Peltolan ajankäyttö viikoittain	53
11.7	Heikki Salon ajankäyttö viikoittain	54

12 Riskit ja niiden seuranta	55
12.1 Riskien todennäköisyydet ja haitat	55
12.2 Muutostarve kehityskoneissa	56
12.3 Jäsenten motivaation puute	56
12.4 Valitun alustan ongelmat	57
12.5 Jäsenten poissaolot	57
12.6 Testipalvelimen ongelmat	58
12.7 Esikäyttäjien aktiivisuus	58
13 Projektiryhmän jäsenten kokemuksia	59
13.1 Mitä tekisimme toisin?	59
13.2 Tero Hännisen kokemuksia omin sanoin	59
13.3 Juho Niemisen kokemuksia omin sanoin	60
13.4 Marko Peltolan kokemuksia omin sanoin	61
13.5 Heikki Salon kokemuksia omin sanoin	62
14 Yhteenveto	63
15 Lähteet	64

1 Johdanto

Jyväskylän yliopistolla ei ole käytössä yhtenäistä tietojärjestelmää lähdekoodien jakamista tai julkistamista varten. Jokainen lähdekoodia kehittävä tutkimusryhmä on joutunut kehittämään omat käytänteensä lähdekoodin levittämiseen ryhmän sisällä ja ulkopuolelle.

Verso-projekti oli Jyväskylän yliopiston tietotekniikan laitoksella keväällä 2010 toteutettu sovellusprojekti. Projekti määritteli, suunnitteli, toteutti ja testasi Jyväskylän yliopiston tietotekniikan laitokselle prototyypin lähdekoodien julkistamisjärjestelmästä. Gitorious-järjestelmään pohjautuvalla YouSource-nimisellä WWW-sovelluksella käyttäjät voivat luoda lähdekoodit sisältäviä tietovarastoja ja julkistaa niitä. Tietovarastoja voi ylläpitää käyttäen hajautetun versiohallinnan työkaluja.

Dokumentissa arvioidaan projektisuunnitelman toteutumista. Projektisuunnitelma [7] määritteli osittain projektin läpiviennin käytänteet, resurssit ja aikataulun. Toteutetun sovelluksen vaatimuksia ja toteutusratkaisujen kartoitustehtäviä ylläpidettiin projektissa Trac-järjestelmässä [30]. Sovelluksen rakennetta ja toimintaa kuvataan sovellusraportissa [16].

Luvussa 2 kuvataan projektiin liittyviä termejä. Luku 3 tarkastelee projektin taustaa ja tavoitteiden toteutumista. Luvussa 4 esitellään projektiorganisaatio ja muita projektin käytössä olleita resursseja. Luvussa 5 kuvataan Verso-projektin yhteistyötä Gitorious-sovellusta kehittävän ja hyödyntävän yhteisön kanssa. Luku 6 käsittelee projektin yleisiä käytänteitä. Luvussa 7 esitellään sovellukseen liittyvää laadunvarmistusta. Luku 8 esittelee projektissa käytetyt versiohallintakäytänteet, ja luvussa 9 esitellään Trac-järjestelmän käytänteet. Luvussa 10 tarkastellaan projektin tehtäviä sekä niiden työmääriä ja jakautumista projektin jäsenten kesken. Luvussa 11 kuvataan Verso-projektissa käytettyä prosessimallia ja projektin aikataulua. Luku 12 kuvaa riskien toteutumista ja hallintaa. Luvussa 13 ryhmän jäsenet kuvaavat kokemuksiaan projektista.

2 Termejä

Luvussa kuvataan dokumentissa esiintyviä termejä aihealueittain.

2.1 Projektin läpivientiin liittyviä termejä

Projektin läpivientiin liittyvät termit ovat seuraavat:

Ketterä ohjelmistokehitys	on joukko käytettäviä menetelmistöjä, joille on yhteistä toimivan ohjelmiston ensisijaisuus, suora viestintä ja nopea muutoksiin reagointi projektin aikana.
Tiketti	(engl. <i>ticket</i>) on yksittäinen ominaisuusvaatimus tai muu tehtäväkuvaus.
Trac	on projektinhallintaohjelmisto ja tikettijärjestelmä, jolla voi muun muassa hallita projektin vaatimusmäärittelyä. Tracin käyttöä on kuvattu tarkemmin luvussa 9.
Vaihe	(engl. <i>phase</i>) on lyhyt ajanjakso projektin toteutuksessa.

2.2 Sovellukseen liittyviä termejä

Projektissa kehitetylle YouSource-ohjelmistolle ja sen pohjana käytetylle Gitorious-sovellukselle ominaisia termejä ovat seuraavat:

Gitorious	on WWW-sovellus Git-tietovarastojen selaamiseen ja hallintaan.
Julkistaminen	tarkoittaa lähdekoodin tai dokumentin asettamista julkisesti saataville.
Lokaali tietovarasto	on paikallinen, vain käyttäjän tietokoneella sijaitseva tietovarasto.

Metatiedostot	ovat tietovaraston sisältöä kuvailevaa lisätietoa, jotka on tallennettu tietovaraston varsinaisesta sisällöstä erotettuihin tiedostoihin.
Palveluprosessi	on tietyn palvelun tarjoava ja taustalla suoritettava ohjelma (engl. <i>daemon</i>).
Projekti	on YouSource-järjestelmässä käyttäjän luoma sisältökokonaisuus, jolla voi olla monia tietovarastoja.
Wiki	on verkkosivukokonaisuus, jonka tavoitteena on helppo muokattavuus ja sivujen välinen vuorovaikutteisuus.

2.3 Git-versiohallintaohjelmiston termejä

Versiohallinnan käyttöä kuvataan luvussa 8. Projektin tulosten hallintaan ja YouSource-sovelluksen kehittämisessä käytetylle Git-versiohallintaohjelmistolle ominaisia termejä ovat seuraavat:

Branch	on versiohistorian haara, committien tietty jatkumo.
Cherry pick	on tietyn commitin poimimista haarasta toiseen.
Commit	on muutos, joka sisältää tiedot muuttuneista tiedostoista, kuvauksen, lähettäjän ja hashin, joka yksilöi commitin.
Hajautettu versiohallinta	tarkoittaa ilman pakollista keskustietovarastoa tehtävää versiohallintaa.
Hash	on kryptografista tiivistefunktio SHA1:ta käyttäen muodostettu tiiviste, jolla Git yksilöi commitit.
Merge request	on pyyntö liittää pyytäjän tekemät ja toimittamat muutokset jonkun toisen ylläpitämään tietovarastoon.
Tietovarasto	(engl. <i>repository</i>) on versioitu tiedon tallennustapa. Dokumentissa viitataan yleensä nimenomaan Git-ohjelmistolla ylläpidettyyn tietovarastoon.

3 Tavoitteiden toteutuminen

Luvussa kuvataan projektin taustaa, tavoitteiden toteutumista ja tuloksia. Projektissa kehitetylle sovellukselle asetetuista tavoitteista ryhmä ehti toteuttaa kaikki paikallisiksi määritellyt vaatimukset sekä lähes kaikki tärkeiksi nähdyt vaatimukset. Myös projektille asetetut oppimistavoitteet toteutuivat.

3.1 Taustaa

Jyväskylän yliopistossa toimii monenlaisia tutkimusryhmiä, joissa toteutetaan lähdekoodia. Yliopistossa edes tietotekniikan laitoksella ei ole yhtenäisiä toimintatapoja tai järjestelmiä lähdekoodin säilyttämiseen, julkistamiseen tai levittämiseen, mistä seuraa monenlaisia ongelmia.

Hajanaisesti tallennettujen lähdekoodien takia tieto lähdekoodista ei leviä välttämättä kehityksen aikana edes ryhmän sisällä, mikä saattaa johtaa päällekkäisen työn tekemiseen. Työntekijän lähtiessä talosta hänen yksin toteuttamansa lähdekoodi käytännössä menetetään, kuten tilaajan alustavassa aihekuvauksessa [27] kuvataan.

Osa tutkimusryhmistä ei käytä ollenkaan versiohallintaa, ja osa vain paikallisesti. Ilman yhteisesti saatavissa olevaa lähdekoodin tallennuspaikkaa itse lähdekoodin jakaminen eri tutkijoiden kesken vaatii jatkuvaa soveltamista, joten kätevin ratkaisu voi lopulta olla lähdekoodin version siirtäminen muistitikulla tai sähköpostilla [13].

Yhtenäisen tallennuspaikan puuttumisen ohella lähdekoodien hyödyntämiseen liittyy oikeudellisia haasteita. Käytäntöjen puuttuessa lähdekoodin yhteyteen ei nimitäin aina tallenneta tietoa sen lisenssistä. Niinpä tutkimusryhmien osaamista on vaikea markkinoida jakamalla lähdekoodia, kun ei ole tietoa, onko lähdekoodin lisenssi yhteensopiva edes yliopiston tutkimusryhmän sisäisesti käytettäväksi.

Projektin tilanneella Jyväskylän yliopiston tietotekniikan laitoksella oli ennen Verso-projektia demokäytössä Redmine-sovellus, jonka avulla oli kerätty tietoa ja kokemuksia kehitettävään tietojärjestelmään liittyvistä tarpeista. Redmine ei kuitenkaan soveltunut sellaisenaan käyttäjille, minkä johdosta muodostui tarve uuden prototyypin kehittävälle sovellusprojektille.

3.2 Sovellukselle asetetut tavoitteet

Tilajana toimineen Jyväskylän yliopiston tietotekniikan laitoksen tavoitteena oli kehittää lähdekoodien julkistamisjärjestelmä. Sen tulisi mahdollistaa sekä koodin julkistaminen maailmalle että myös levittää tietoisuutta kehitetyistä ohjelmista oman yliopiston sisällä. Kehitettävän sovelluksen tavoitteena oli myös mahdollistaa yksityisten tietovarastojen luominen, jotta järjestelmää voisi käyttää ilman tarvetta julkistaa kaikkea. Projektin ja sovelluksen tavoitteena oli myös yleisesti kannustaa aloittamaan versiohallinnan käyttöä.

Traciin kirjattujen ohjelmistoon liittyvien vaatimusten ja tehtävien määrät esitetään taulukossa 3.1. Tärkeimmät tilaajan tavoitteet kohdistuivat käytön helppoon aloittamiseen ja yksityisten tietovarastojen mahdollistamiseen. Projektin kuluessa järjestelmälle määritettiin myös monia pienemmän prioriteetin saaneita vaatimuksia.

Prioriteetti	Toteutetut	Toteuttamatta	Yhteensä
Mandatory	11	0	11
Important	4	2	6
Useful	5	8	13
Left out	1	1	2

Taulukko 3.1: Tracin tikettien määrät prioriteeteittain.

Gitorious tarjosi sovellukselle asetetuista vaatimuksesta seuraavat:

- tietovaraston luonti ja hallinta WWW-käyttöliittymän kautta
- käyttöoikeuksien hallinta WWW-käyttöliittymän kautta
- tietovarastojen selaaminen ja niistä hakeminen
- sivuston yleisen aktiviteetin esittäminen sekä
- tietovarastokohtaisen kuvauksen asettaminen.

Projektiryhmä toteutti sovellukseen uusina toimintoina tai muokkasi huomattavasti Gitoriuksen osalta seuraavia:

- sovelluksen ajossa olevan version tunnisteiden näyttäminen käyttäjille
- Korppi-opintotietojärjestelmän käyttäjätunnuksilla kirjautuminen
- projektien ja tietovaraston näkyvyyksien asteittainen rajoittaminen

- ohjattu uuden tietovaraston luonti
- selausnäkyä tietovarastoille
- tietovaraston metatietojen tallentaminen tietovarastoon
- tietovaraston päivittäminen tiedoston tai paketin antamalla
- tietovaraston asettaminen peilaamaan ulkoista SVN-tietovarastoa tai pakattua tiedostoa
- lisenssien valitseminen tietovarastokohtaisesti projektikohtaisuuden sijaan
- projektin wikin käyttämisen mahdollistaminen projektin muiden tietovarastojen tapaan sekä
- yleisiä käytettävyyssparannuksia.

Toteutettu järjestelmä kattaa tärkeimmät siihen kohdistuneet vaatimukset. Projektin tuloksena on kehityskelpoinen järjestelmä, jota voi jo sellaisenaan käyttää lähdekoodien julkistamiseen ja työkaluna ohjelmoinnin aikana. Sovelluksen toteuttamisratkaisuja kuvataan tarkemmin sovellusraportissa [16].

Käyttöliittymän osalta Gitoriousiin ei tehty merkittäviä rakenteellisia muutoksia, mutta projektin aikana käyttöliittymiin tehtiin useita projektin aikana ilmenneisiin parannusehdotuksiin perustuvia käytettävyysskorjauksia. Uusien ominaisuuksien myötä Gitoriousin WWW-käyttöliittymään lisättiin myös muutama täysin uusi sivu.

3.3 Tavoitteiden muutokset projektin kuluessa

Suurin osa tärkeimmistä vaatimuksista oli selvillä projektin alusta asti. Alustaksi valittu Gitorious kuitenkin aiheutti projektin edetessä pakolliseksi määriteltyjä muutostarpeita. Yksi tällainen muutos oli tietovarastojen selaaminen, koska Gitorious mahdollisti selaamisen vain projekteille, jonka alaisuudessa tietovarastot Gitoriousissa ovat.

Käytön aloittamisen helpottamiseksi muun muassa tietovaraston luontia Gitorioussovelluksessa nopeutettiin muokkaamalla olemassaolevia toimintoja. Lisäksi sovellus yhdistettiin 19.4.2010 käyttämään kirjautumisessa Jyväskylän yliopiston Korppiopintotietojärjestelmän käyttäjätunnuksia, mikä poisti kokonaan käytön aloituksessa vaaditun erillisen rekisteröitymisen.

3.4 Sovelluksen jatkokehitys

Projektissa toteutettu YouSource-sovellus oli luonteeltaan prototyyppi, jonka kehittämiseksi kerättiin palautetta myös projektiorganisaation ulkopuolisilta esikäyttäjiltä. Jo projektin alkaessa oli selvää, ettei kaikkia ideoituja toiminnallisuuksia ehditä toteuttamaan Verso-projektin aikana.

Tilaajan kanssa sovittiin jatkokehitykseen tavoitteina olleista ominaisuuksista seuraavat:

- autentikoituminen Kerberosta käyttäen tietovarastojen päivittämisen yhteydessä
- selaimen kautta tapahtuva tietovaraston päivittäminen ilman käyttäjälle lisättyä SSH-avainta
- tietovaraston yksittäisen version julkistaminen versiohallinnan kautta tallennetuista versioista
- merge requestien hyväksyminen WWW-käyttöliittymällä
- tietovaraston päivittäminen sähköpostilla
- lähdekooditiedostojen kommentointimahdollisuus
- lähdekoodin sertifiointi sekä
- tietovarastokohtaiset kategoriat (engl. *tags*).

Verso-projekti toteutti YouSource-järjestelmään kaikki pakollisiksi määritellyt ja lähes kaikki tärkeiksi määritellyt ominaisuudet. Suurin osa jatkokehitykseen sovituisista ominaisuuksista valittiin pieneksi määritellyn prioriteetin takia, mutta esimerkiksi Kerberos-autentikointi katsottiin liian suuritöiseksi toteutettavaksi projektin lopussa.

Toteutetut ja toteuttamattomat ominaisuudet sekä ominaisuuksiksi hyväksymättömät ideat on kirjattu Trac-järjestelmään [30] ja sovellusraporttiin [16]. Trac-tiketit toimivat mahdollisena pohjana jatkokehitykselle. Projektiryhmästä Marko ja Tero jatkoivat Verso-projektin loputtua jatkokehittämään sovellusta tietotekniikan laitokselle.

3.5 Projektin tulokset

Projektin kuluessa toteutetut tulokset voidaan jakaa sovellukseen ja projektiin liittyviin tuloksiin. Projektin tulokset on tallennettu seuraaviin kahteen tietovarastoon:

Projektitietovarasto	sisältää sekä projektiin että sovellukseen liittyvät dokumentit ja tiedostot sekä niiden Git-versiohallinnan.
Sovellustietovarasto	sisältää YouSource-prototyypin lähdekoodin sekä Git-versiohallinnan.

Projektiryhmä toteutti seuraavat julkiset suunnitelmat ja raportit:

Ajankäyttöraportti	sisältää ryhmän jäsenten kirjaamat työtunnit.
Asennusohje	sisältää YouSourcen asennusohjeet.
Esikäyttösuunnitelma	kuvaa esikäyttäjien hyödyntämistä projektissa ja käytettävyydestestauksien läpivientä.
Esittelymateriaalit	sisältävät väli- ja loppuesittelyn esitysgrafiikan ja muistiot.
Järjestelmätestaussuunnitelma	määrittelee sovelluksen toimivaksi toteamiseen vaaditut testitapaukset.
Järjestelmätestausraportit	kuvaavat kehitetyn sovelluksen järjestelmätestauskertojen tulokset.
Käytettävyysmuistiot	sisältävät kahden käytettävyystestauksen muistiot.
Projektiraportti	kuvaa projektin läpivientä ja asetettujen tavoitteiden saavuttamista.
Projektisuunnitelma	kuvaa projektin tavoitteita, tehtäviä, aikataulua, yleisiä käytänteitä ja riskien hallintaa.
Sovellusraportti	kuvaa toteutetun sovelluksen toteutusratkaisut, osat ja toiminnot sekä jatkokehitysideat.

Projektisuunnitelma jäädytettiin 12.4.2010, koska suunnitelman tekemisen ei enää katsottu edistävän projektin läpiviemistä. Projektisuunnitelmassa oli valmiina Trac- ja muiden käytänteiden kuvaus, projektin tavoitteet tuloksien ja oppimistavoitteiden osalta sekä kuvaus projektissa käytettävästä prosessimallista. Projektisuunnitelmasta puuttui projektissa valmistettavan sovelluksen kuvaaminen, sekä siinä oli vasta alustavasti kuvattu riskejä ja tuntien jakautumista.

Edellä mainittujen dokumenttien lisäksi ryhmä laati seuraavat tulokset:

Alustavertailu	sisältää projektin alussa laaditun vertailun potentiaalisista alustavaihtoehdoista.
Itsearviointit	sisältävät ryhmän jäsenten arviointit omasta panoksesta, onnistumisesta ja oppimisesta.
KorppiLDAPin käyttökuvaus	kuva, miten projektissa toteutettu sovellus käyttää KorppiLDAPia.
Lisenssisopimus	sisältää jäsenten suostumuksen projektissa toteutetun ohjelmakoodin sijoittamisesta AGPLv3-lisenssin alaisuuteen.
Lähdekoodi	sisältää toteutetun lähdekoodin kommentteineen.
Palaverien dokumentit	sisältävät palaverien esityslistat ja pöytäkirjat.
Palavereiden esityskalvot	sisältävät palavereissa käsitellyt asiat tilakatsauksineen 4. palaverista alkaen.
Sähköpostiarkisto	sisältää kaikki projektin sähköpostilistan viestit, mukaanlukien päivittäiset tilakatsaukset.

3.6 Oppimistavoitteet

Sovellusprojekti on tietotekniikan syventävien opintojen kurssi, jossa neljästä opiskelijasta muodostettu ryhmä toteuttaa tilaajalle ohjelmiston tietotekniikan laitoksen ohjauksessa. Kurssissa tutustutaan projektimuotoiseen toimintatapaan. Sen tärkeimpiin tavoitteisiin kuuluu antaa ryhmän jäsenille kokemusta ryhmätyöstä, projektin läpiviennistä sekä erilaisista käytänteistä kirjallisessa ja suullisessa viestinnässä.

Koko ryhmä halusi projektin alkaessa oppia projektin hallintaa ja läpivientiä. Lisäksi Juho, Tero ja Marko halusivat projektin aikana saada lisää kokemusta ohjelmoinnista sekä oikeassa ohjelmistoprojektissa mukanaolosta ja versiohallinnan käyttämisestä. Heikki halusi saada kokemusta projektin läpiviennistä, minkä johdosta hänet valittiin projektipäälliköksi. Kaikki projektiryhmän Verso-projektille asettamat oppimistavoitteet täyttyivät.

Oheiskurssien opetustapahtumissa Verso-ryhmä sai koulutusta projekti- ja ryhmätyöskentelystä, projektin johtamisesta ja hallinnasta, tekijänoikeuksista, sopimuksista ja käytettävyydestä.

3.7 Tekniikoista ja ohjelmistoista opittua

Verso-projektin jäsenet saivat Gitorious-sovellusta jatkokehittäessään kokemusta Ruby-ohjelmointikielestä, Ruby on Rails -ohjelmistokehityksestä sekä WWW-sovelluksen ylläpitämisestä. Tietokantana oli MySQL 5.0 -tietokannanhallintajärjestelmä, ja projektin aikana ryhmän jäsenet oppivat muun muassa SQL-kieltä.

Projektiryhmä oppi projektin aikana hyödyntämään hajautettua versiohallintaa käyttämällä Git-versiohallintaohjelmistoa sekä sovelluksen kehittämiseen että projektin dokumenttien hallintaan. Ryhmä oppi myös avoimen lähdekoodin ohjelmistoprojekteihin liittyviä versiohallintakäytänteitä. Lisäksi oman sovelluksen käyttämiselle ja esikäyttöön tarjoamiselle projektin aikana edellytys oli, että ryhmä oppi kehittämään keskeneräisiä ominaisuuksia omissa haaroissaan versiohallinnassa, jolloin päähaara oli jatkuvasti otettavissa käyttöön.

Projektin vaatimuksia, kehitysideoita sekä löydettyjä virheitä ylläpidettiin projektin ajan Trac-projektinhallintasovelluksessa. Projektin aikana ryhmä oppi perustaidot yleisen projektinhallintatyökalun käytöstä.

Projektin aikana ryhmän jäsenet oppivat käyttämään Linux-työpöytäympäristöä ja esikäyttöä varten hankittua Linux-virtuaalipalvelinympäristöä. Projektiryhmä oppi ylläpitämään virtuaalipalvelinta sovellukseen liittyvien palveluiden osalta itsenäisesti. Ryhmä oppi myös toimimaan tietohallintokeskuksen ATK-käytänteiden kanssa halutessaan muutoksia työpöytäkoneille esimerkiksi asennettavien sovellusten osalta.

4 Organisaatio ja resurssit

Luvussa esitellään projektiorganisaatioon kuuluneet henkilöt, ryhmän käytössä olleet tilat, laitteet ja ohjelmistot sekä jäsenille järjestetyt perehdytykset.

Projektin aloittamisen jälkeen tärkein poikkeama resursseissa on Paavo Niemisen mukaantuleminen tietotekniikan laitoksen neljänneksi edustajaksi. Organisaation muuttuminen ei vaikuttanut projektin kulkuun.

4.1 Projektiorganisaatio

Verso-projektiryhmään kuului neljä Jyväskylän yliopiston tietotekniikan opiskelijaa. Tero Hänninen oli projektin alkaessa 3. vuoden ja Marko Peltola 7. vuoden tietotekniikan opiskelija. Juho Nieminen ja Heikki Salo olivat 4. vuoden tietotekniikan opiskelijoita. Opiskelijoista Heikillä oli kokemusta WWW-sovellusten kehittämisestä sekä ohjelmistoprojektissa toimimisesta Jyväskylän yliopiston opintotietojärjestelmä Korpin kehityksen kautta. Ryhmällä ei ennen projektia ollut kokemusta projektin läpiviennistä, eikä suuresta osasta kehitysympäristöä. Kaikilla ryhmän jäsenillä oli vahvuutena hyvät sosiaaliset taidot sekä erinomaiset valmiudet omaksua uusia työkaluja ja tekniikoita.

Tilajana toimineen tietotekniikan laitoksen edustajina toimivat Paavo Nieminen, Tapani Tarvainen, Ville Tirronen ja Tero Tuovinen, joista Tirrosella oli määräysvalta tilaajaa koskevissa asioissa. Ryhmän vastaavana ohjaajana toimi Jukka-Pekka Santanen, ja teknisenä ohjaajana toimi Antti-Juhani Kaijanaho. Paavo Nieminen liittyi projektiin tilaajan edustajaksi projektin neljännessä palaverissa [18]. Muita muutoksia projektiorganisaatioon ei projektin aikana tullut.

Projektiryhmän työkoneiden ohjelmistojen ja ylläpidon hoiti Jyväskylän yliopiston ATK-lähituesta pääasiassa Santeri Lapinmäki. Projektiryhmän käytössä olleen virtuaalipalvelimen ylläpidosta vastasi Jyväskylän yliopiston tietohallintokeskuksen sovelluspalveluista pääasiassa Harri Tuomi.

Projektin aikana suoritettujen puhe- ja kirjoitusviestinnän kurssin osalta projektiryhmää auttoivat kurssin opettajat Leena Peltomaa (kirjoitusviestintä) ja Minna Koljonen (puheviestintä). Käytettävyyssasioissa projektiryhmää auttoi Meeri Mäntylä, joka piti ryhmälle käytettävyysspäivän.

Graafinen suunnittelija Auri Kaihlavirta laati sovellukselle logokuvan ja määrittä värit, joiden avulla sovellukselle saatiin persoonallinen ilme.

Projektissa toteutettua sovellusta tarjottiin esikäyttöön. Projekti tekikin kevään aikana yhteistyötä parinkymmenen tietotekniikan laitoksen henkilökuntaan ja opiskelijoihin kuuluvan esikäyttäjän kanssa.

4.2 Projektin tilat ja laitteet

Tilojen ja laitteiden osalta toteuma vastasi täysin suunniteltua. Projektiryhmän käytössä ollut huone AgC222.2 sijaitsi Agoran C-siivessä toisessa kerroksessa sovellusprojektien tiloissa. Huone tarjosi kevään ajan työskentelytilan projektiryhmälle sisältäen työasemat, työpöydät, tuolit, naulakon, kirjahyllyn sekä kaksi tussitaulua.

Huoneessa oli ryhmän käytössä neljä tietokonetta, joista kolmessa oli käyttöjärjestelmänä Linux Fedora Core 12 ja yhdessä Windows XP SP3 ryhmän jäsenten työtuntien kirjaamista varten. Projektiryhmällä oli lisäksi verkon kautta käyttöoikeus myös erilliseen testipalvelimeen `versotest.it.jyu.fi`, jonka käyttöjärjestelmänä oli Red Hat Enterprise Linux 5.0.

Projektiryhmän käytössä oli lisäksi sovellusprojektien kannettava tietokone, videoprojektorin ja sanelin sekä projektitilojen monitoimilaite. Projektiryhmän käytössä oli myös kokoustila AgC222.6, jossa pidettiin kaikki projektipalaverit.

4.3 Ohjelmointi- ja dokumentointityökalut

Projektissa käytetyt ohjelmointi- ja dokumentointityökalut vastasivat suunniteltuja lähes täysin. Suunnitelmassa ei tarkasti otettu kantaa koodin ja raakatekstin kirjoittamiseen käytettävistä ohjelmista, joten jäsenet käyttivät ja kokeilivat projektin edetessä useita erilaisia työkaluja.

Linux-tietokoneisiin oli Gitorious-sovelluksen kehittämistä varten asennettu MySQL 5.0 -tietokantaohjelmisto, Apache -WWW-palvelin, Ruby-ohjelmatulkki sekä Ruby on Rails -sovelluskirjasto. Windows XP -tietokoneeseen oli ATK-tuelta pyydetty TortoiseGit-ohjelmisto versiohallintaan.

Projektiryhmä ei käyttänyt ohjelmointiin raskaita ohjelmointityökaluja- tai ympäristöjä. Ohjelmointi ja dokumentointi suoritettiin pääasiassa käyttäen Notepad++-

ohjelmaa Windows XP -työasemassa sekä Gedit- ja Vim-ohjelmia Fedora Core -työasemissa).

Työmäärien kirjaamiseen käytettiin Petri Heinosen kehittämää Excel-pohjaista ajankäytönseurantasovellusta [20]. Ladottujen dokumenttien ja palaverien tukena käytetyn esitysgrafiikan valmistamiseen projektiryhmä käytti L^AT_EX-ohjelmistoa.

Projektioorganisaation käytössä oli myös Trac-sovellus, jota käytettiin sovelluksen vaatimusten ja projektin tehtävien hallintaan. Tracin käytössä noudatettuja käytänteitä on tarkasteltu luvussa 9.

4.4 Luennot ja perehdytykset

Toteutuma vastasi luentojen ja perehdytysten osalta suunniteltua. Projektissa vaadittiin hyvää tuntemusta Git-versiohallinnasta, joka ei ollut jäsenille entuudestaan tuttu. Muista sovellusprojekteista poiketen Versolle ei erikseen järjestetty versiohallinnan toiminnasta kertovaa luentoa, vaan järkevämmäksi katsottiin, että tekninen ohjaaja Antti-Juhani Kaijanaho oli käytettävissä asiantuntijana versiohallintaan liittyvissä kysymyksissä ja järjesti ryhmälle ongelmalähtöistä perehdytystä.

Projektissa toteutetun prototyypin ohjelmointiin tarvittiin tuntemusta Rubystä, Ruby on Railsista sekä sovelluksen pohjana toimivan Gitoriuksen toiminnasta. Projektiryhmä perehtyi Rubyyn ja Ruby on Railsiin omatoimisesti tutkimalla Internetissä saatavilla olevia ilmaisia oppaita. Lisäksi ryhmä tilasi Santasen kautta käyttöönsä Agile Web Development with Rails -kirjan [21]. Gitoriuksen asentamiseen ja toimintaan projektiryhmä tutustui lähdekoodeihin perehtymällä sekä kysymällä apua Gitoriuksen kehitysyhteisöltä sähköpostilla ja IRC-kanavalla.

Oheiskurssilla opettivat projektin vastaava ohjaaja Jukka-Pekka Santanen, kirjoitusviestinnän opettaja Leena Peltomaa, puheviestinnän opettaja Minna Koljonen, tietotekniikan opettaja Antti-Juhani Kaijanaho, sekä käytettävyyssiantuntija Meeri Mäntylä. Oheiskurssilla läpikäytyihin asioihin kuuluivat

- esittely ja esiintyminen (Koljonen)
- kirjoitusviestintä (Peltomaa)
- kokous- ja neuvottelukäytänteet (Koljonen)
- käytettävyyden luennot ja ryhmätyöt (Mäntylä)
- projektin johtaminen ja hallinta (Santanen)
- tekijänoikeus ja sopimukset (Santanen) sekä

- versiohallinta (Kaijanaho).

Projektiviestintä IT-alalla -oheiskurssiin kuului sekä kirjoitus- että puheviestintää. Kirjoitusviestinnässä projektiryhmä paneutui virallisten dokumenttien kieli- ja ulkoasuvaatimuksiin. Puheviestinnässä projektiryhmä harjoitteli kommunikointitaitoja ryhmän keskinäistä viestintää sekä esitystilanteita varten. Esitystilanteisiin valmistauduttiin pitämällä kaksi väliesittelyä, joissa projektiryhmä harjoitteli esiintymistä projektin loppuesittelyä varten.

Oheiskurssin *Sovellusprojektin viestintää, hallintaa ja työkaluja* opetussisältöön sisältyi katsaus sovellusten käytettävyyteen. Meeri Mäntylä piti projektiryhmälle käytettävyysspäivän, jossa käytiin läpi yleisiä käytettävyyteen liittyviä seikkoja sekä harjoiteltiin käytettävyydestauksen pitämistä. Lopuksi Mäntylä koekäytti sovellusta ja kertoi huomioitaan, joista kirjattiin muistio [22]. Jukka-Pekka Santanen piti luennon projektin hallinnasta ja läpiviennistä. Luennolla käsiteltiin myös ryhmän jäsenten keskinäistä viestintää.

5 Yhteistyö Gitorious-yhteisön kanssa

Projektin tavoitteena oli toteuttaa alkuperäisen Gitoriousin kanssa yhteensopivia ominaisuuksia, joita yritettäisiin saada alkuperäiseen Gitorious-ohjelmistoon ja/tai käyttöön muille kehittäjille. Projektiryhmä kommunikoi projektin aikana Gitorious-yhteisön kanssa pääasiassa Freenode-verkossa #gitorious-IRC-kanavan sekä Gitorious-sähköpostilistan [4] avulla.

Projektisuunnitelma [7] rajoittui kuvaamaan Gitorious-yhteisön kanssa toimimista vain ohjelmiston osalta. Suunnitelmasta poikettiin teknisten ongelmien vuoksi, jotka hankaloittivat ominaisuuksien saamista käyttöön YouSourcen ulkopuolella. Ominaisuuksien julkaisua on kuvattu luvussa 5.2.

5.1 Ongelmien ratkaiseminen

Gitorious-sovellus sisältää monia ohjelmakokonaisuuksia, ja sen asentaminen sisältää monia vaiheita. Myös sovellukseen liittyvien virhetilanteiden selvittäminen oli välillä haastavaa. Gitorious-sovellukseen liittyvissä ongelmatilanteissa projektiryhmä yritti ensin itse ratkaista ongelman lähdekoodia tai Internetistä löytynyttä dokumentaatiota tutkimalla. Hankalissa tapauksissa projektiryhmä kysyi apua #gitorious-kanavalta (IRC-verkossa *freenode*), josta saatiin usein tarvittava apu.

Asennus- ja ohjelmointipulmissa kysymykset vähentyivät projektin alun jälkeen projektiryhmän oppiessa sovelluksesta ja sen ympäristöstä lisää. Projektiryhmä kevään kuluessa myös neuvoi #gitorious-kanavalla muita.

5.2 Tulosten julkistaminen yhteisölle

Projektin tavoitteena oli alusta asti julkaista projektissa toteutettuja osia muulle yhteisölle. YouSource-sovellus on Gitoriousin pohjautuessaan jo lisenssinsä puolesta avointa lähdekoodia, mutta avoimuuden lisäksi Gitoriousin päälle rakennettavat muutokset pidettiin mahdollisimman yhteensopivina alkuperäisen järjestelmän kanssa. Alkuperäisen järjestelmän henkeen toteutetuilla ja yhteensopivilla komponenteilla on parhaat mahdollisuudet päätyä osaksi alkuperäistä järjestelmää, ns. Gitorious-mainlineä.

Alkuperäisen ohjelmiston osaksi saadut mahdolliset muokkaukset hyödyttävät kaikkia osapuolia. Etenkin Verson toteuttamien ominaisuuksien saamisesta Gitoriukseen hyötyy ne kehittänyt tilaaja, sillä siten ominaisuuksien ylläpito ei ole enää yksin alkuperäisen kehittäjän vastuulla. Lähtökohtaisestikin tavoitteena oli saada käyttöön mahdollisimman hyvin tarvetta vastaava järjestelmä.

Verso-projekti lähetti Gitorious-alustaan muokkauspyyntöinä (engl. *merge request*) bugikorjauksia, joista projektin aikana varsinaiseen Gitorious-sovellukseen Gitoriouksen pääkehittäjä Johan Sørensen hyväksyi kolme. Lähetetyt korjaukset sisälsivät korjaukset käyttäjäprofiiliin liittyvän avatar-kuvan poistoon, automaattisesti täydentyvien tekstikenttien korjaukset sekä salasanojen suodattamisen sovelluksen poikkeusten yhteydessä lähetyistä virheviesteistä.

Kehitettyjä ominaisuuksia ja bugikorjauksia lähetettiin muun yhteisön saataville vasta toukokuussa, minkä takia versiohallintaan liittyvissä käytänteissä havaitut puutteet ilmenivät vasta silloin. Versiohallintaan ja julkaisemiseen liittyviä ongelmia on kuvattu tarkemmin luvussa 8.2.

Projektin jäsenet lähettivät lisäksi sähköpostitse huomioita Gitorious-sovelluksen parantamiseksi. Esimerkki onnistuneesta vuorovaikutuksesta oli Gitoriouksen mainline-version ulkoasun vaihduttua 18.5.2010, kun jäsenet esittivät sekä IRC-kanavalle että postilistalle parannusehdotuksia siihen. Ulkoasun kehittänyt Ole Martin Kristiansen kiitti ehdotuksista ja kertoi toteuttaneensa yhden ja kirjanneen loput kehitettäväksi [29].

6 Käytänteet

Luvussa kuvataan projektin suunniteltuja käytänteitä ja niiden toteutumista. Käytänteillä pyrittiin ennenkaikkea varmistamaan tulosten ja projektin laatua. Sovellukseen keskittyvää laadunvarmistusta on kuvattu luvussa 7. Versiohallintaan liittyvät käytänteet ovat laajuutensa takia kuvattu luvussa 8.

Projektisuunnitelmassa [7] kuvatut käytänteet toteutuivat lähes sellaisinaan, eikä niistä poikettu tietoisesti. Suurin poikkeama ovat muutamat lähettämättä jääneet päivittäiset tilakatsaukset, mikä johtui unohduksesta tai huonosta tiedonkulusta. Päivittäisiä katsauksia jäi lähettämättä kuitenkin vain puolenkymmentä, mistä ei aiheutunut haittaa tai muutoksia projektin kulkuun.

6.1 Tiedotus

Toteutuneet tiedotuskäytänteet vastasivat hyvin suunniteltua. Projektin edetessä suurimmat muutokset olivat lähinnä hienosäätöä, kuten Trac-järjestelmän muutoksien ilmoitusviestien käyttöönotto.

Projektiryhmän tiedotuksesta vastasi käsillä olevasta asiasta vastaava projektiryhmän jäsen tai projektipäällikkö. Lähtökohtaisesti tiedotuksen hoiti projektipäällikkö, joka tarvittaessa nimesi toisen ryhmän jäsenen hoitamaan tietyn asiakokonaisuuden tiedottamista. Esimerkiksi kehitystehtäviin liittyvät yhteydenotot projektissa teki pääsääntöisesti kehittäjä itse.

Projektiryhmän jäsenten, tilaajan ja ohjaajien välinen tiedotus hoidettiin sähköpostilla käyttämällä sähköpostilistaa `verso@korppi.jyu.fi`. Atk-tuelle ja sovelluspalveluille projektin aikana lähetetyt palvelupyynnöt sekä niiden vastaukset ohjattiin projektiorganisaatiolle postilistan kautta. YouSource-sovelluksen esikäyttäjien keskusteluun ja kysymyksille perustettiin sähköpostilista `yousource-users.group@korppi.jyu.fi`.

Projektiryhmän jäsenten välinen tiedotus hoidettiin sähköpostia ja IRC-kanavaa käyttämällä. Jäsenillä oli eri sähköpostilistojen lisäksi oma sähköpostilista keskinäiseen viestinvaihtoon.

Projektin aikana projektiryhmä lähetti päivittäin tilakatsauksen kuluneesta päivästä. Katsaus sisälsi henkilöittäin koosteen tehdyistä asioista, vastaantulleista ongel-

mista sekä seuraavaksi toteutettavista asioista. Päivittäisen katsauksen lähetti projektipäällikkö tai projektipäällikön nimeämä jäsen projektipäällikön ollessa estynyt.

Projektin ajan Trac-järjestelmän muutoksista ja lisätyistä tiketeistä ohjautui projektiorganisaation sähköpostilistalle automaattinen ilmoitus. Tracin käyttöä projektissa on kuvattu tarkemmin luvussa 9.

6.2 Palaverit

Projektin alussa sovitut palaverikäytännöt toteutuivat projektissa muutoksitta. Koska palaverien lukumääräksi sattui lopulta tasan 10, kukin jäsen oli tasan kahdesti sekä puheenjohtajan että sihteerin roolissa palaverissa.

Projektipalavereja pidettiin 1–2 viikon välein. Projektipalavereihin osallistuivat ryhmän jäsenet, ohjaajat ja tilaajan edustajat. Ensimmäisessä palaverissa [13] sovittiin palaverin olevan päätösvaltainen, kun projektiryhmästä ja tilaajista oli paikalla vähintään yksi edustaja sekä ohjaajista vastaava ohjaaja Santanen. Tilaajalla oli myös mahdollisuus tarvittaessa nimittää ulkopuolinen edustaja palaveriin. Palaverit olivat laillisia, kun palaverikutsu ja esityslista oli lähetetty vähintään vuorokautta ennen palaveria.

Palaverit avasi edellisen palaverin puheenjohtaja, minkä jälkeen projektiryhmä ehdotti keskuudestaan puheenjohtajaa ja sihteerää. Projektiryhmä kierrätti puheenjohtajan ja sihteerin rooleja siten, että kukin toimi projektin aikana kahdesti puheenjohtajana ja sihteerinä. Palavereissa asiat käytiin läpi kokouskutsun mukana lähetetyn esityslistan mukaisesti, johon kokoukseen osallistujilla oli mahdollisuus esittää tarvittaessa lisäyksiä esityslistaa hyväksyttäessä.

Toteutusvaiheiden ajan palavereiden tärkein sisältö oli edellisen toteutusvaiheen tuloksien hyväksyminen ja seuraavassa vaiheessa toteutettavien toiminnallisuuksien valitseminen ja kiinnittäminen.

Edellisen palaverin pöytäkirja oli lähetetty sähköpostilistalle tarkistettavaksi etukäteen, joten palavereissa riitti käydä läpi edellinen palaveri vain päätöksiä osalta. Edellisen palaverin pöytäkirja hyväksyttiin tai se hyväksyttiin muutoksin. 4. palaverista lähtien palavereiden runkona oli projektiryhmän laatima kalvoesitys, joka heijastettiin videoprojektorilla nähtäville.

Palavereiden jälkeen sihteeri laati palaverista pöytäkirjan, joka oli ensin projektiryh-

män tarkastettavana. Tämän jälkeen sihteeri lähetti sen projektiorganisaatiolle sekä kirjoitusviestinnän opettaja Leena Peltomaalle hyväksyttäväksi.

6.3 Dokumentoinnin käytänteet

Sovellukseen liittyvät dokumentit kirjoitettiin englanniksi ja projektiin liittyvät dokumentit suomeksi. Englanninkielisissä dokumenteissa käytettiin amerikanenglantia, kuten projektin ensimmäisessä palaverissa sovittiin [13].

Tilaaajan toivomuksen mukaan projektin dokumentit kirjoitettiin pääsääntöisesti raakatekstiksi, jotta ne ovat kevyitä lukea ja jotta ne versioituvat versiohallinnassa paremmin. Oppimisen vuoksi ohjaaja Jukka-Pekka Santanen edellytti projekti- ja sovellusraporttien latomista \LaTeX -ohjelmistoa käyttäen [13].

6.4 Julkistettujen tulosten versionumerointi

Dokumenteissa käytettiin kolmitasoista versionumerointia ensimmäisessä palaverissa [13] sovitun mukaisesti. Ensimmäinen ryhmän jäsenten ulkopuolelle julkistettu versio oli 0.1. Ryhmän sisäiseen käyttöön tarkoitettuja versionumeroita kasvatettiin 0.0.1:llä, kun ohjaajalle, kirjoitusviestinnän opettajalle ja tilaajalle versionumeroita kasvatettiin 0.1:llä. Dokumentin ensimmäinen hyväksytty versio oli 1.0.0. Dokumenttien versionumerointi toteutui suunnitellusti.

Sovelluksen versionumerointi poikkesi suunnitellusta. Sovelluksen versionumeroinnin sovittiin projektin alussa kasvavan nopeasti siten, että sovelluksen julkistamishetkellä versionumero olisi 1.0 ja kasvaisi uuden ominaisuuden tullen yhdellä [13]. Santanen ehdotti dokumentoinnin vuoksi käytettävän lisäksi sivussa versionumerointia, joka alkaisi 1.0:sta ja päättyisi 2.0:aan, sillä sovellusprojekteissa käytetään yleensä vastaavaa numerointia. Versionumeroille ei kuitenkaan ollut sisäistä tai ulkoista käyttötarkoitusta, koska versioiden yksilöinti tehtiin tarvittaessa tietovaraston committien hasheja käyttämällä.

Sovelluksen osalta versiot projektin alussa ja lopussa on kiinnitetty tietovaraston versihistoriaan nimettyinä committeina, *tageina*. Projektin alun ja lopun tagit ovat `verso-project-start` ja `verso-project-final`.

6.5 Luovutettujen tulosten hakemistorakenne ja tiedostojen nimeäminen

Kesken jäänyt projektisuunnitelma [7] ei määritellyt tuloksen luovuttamiseen käytettävää hakemistorakennetta, sillä projektin alussa tavoitteeksi asetettiin tulosten julkistamisen tapahtuvan projektissa toteutettavalla sovelluksella. Verso-projektin tuloksista sovittiin kuitenkin koottavan myös CD:t ja WWW-sivu, joiden toteutunutta rakennetta ei siis voi verrata suunniteltuun.

Sovelluksen lähdekooditietovarasto sekä dokumenttitiedostot ovat projektin WWW-sivuilla sekä CD-levyllä siten, että kansioden `app` (sovellus) sekä `docs` (dokumentit) alta löytyy Git-tietovarastojen työkansiot (engl. *working dir*).

Sovellukseen liittyvä dokumentaatio on pääosin englanniksi, ja projektiin liittyvä suomeksi. Allaolevasta hakemistorakenteesta on tiiviin esitysmuodon saamiseksi karsittu tarpeettomia (merkitty kolmella pisteellä . . .). Hakemistorakenne WWW-sivuilla ja CD:llä on seuraava:

```
.
|-- app
|   |-- ...
|-- docs
|   |-- application
|   |   |-- application_report
|   |   |-- pics
|   |   |-- requirement_specification
|   |   |-- solutions
|   |   |-- system_test_plan
|   |   `-- wishlist
|   `-- project
|-- esittelyt
|   |-- loppuesittely
|   |-- valiesittely1
|   `-- valiesittely2
|-- katselmoinnit
|-- kaytettavyystestaukset
|-- palaveriesitykset
|   `-- ...
```



```
|-- palaverit  
|-- projektiraportti  
`-- projektisuunnitelma
```

Tiedostot kansioissa on nimetty tuloksen nimen mukaisesti välilyönnit alaviivoilla (engl. *underscore*) korvaten ja ääkköset aakkosiksi vaihtaen. Dokumenttitiedostojen tiedostonimissä ei ole versionumeroita lukuunottamatta niistä julkistettuja PDF-versioita.

6.6 Tulosten koostaminen ja toimittaminen

Verso-projektin kehitystyö sekä tulostiedostojen että ohjelmiston osalta on näkyvillä projektin päätyttyessä osoitteissa

- <http://sovellusprojektit.it.jyu.fi/verso/> sekä
- <http://versotest.it.jyu.fi/verso/> (testipalvelin).

Projektin aikana syntynyt **sähköinen materiaali tallennettiin lisäksi CD:lle**, josta toimitettiin kappaleet tilaajalle, projektikansioon ja tietotekniikan laitoksen arkistoon. Sähköinen materiaali pitää sisällään luvussa 3.5 mainittujen tulosten uusimmat versiot, tietovarastojen versiohistorian, Trac-järjestelmän tietoineen ja sähköpostiarkistot. CD:ltä löytyvä sähköinen materiaali on saatavilla myös Verso-projektin sivuston <http://sovellusprojektit.it.jyu.fi/verso/> kautta.

Projektin tuloksista on koostettu **kansio**. Lisäksi kansiossa on kappale allekirjoitusta lisenssisopimuksesta, projektin väliesittelyistä saadut palautteet, jäsenten it-searvioinnit sekä vastaavan ohjaajan ja tilaajan lausunnot.

7 Sovellukseen liittyvä laadunvarmistus

Projektiin ei osin sen prototyypiluonteen vuoksi ja osin kesken jääneen projektisuunnitelman vuoksi suunniteltu kattavaa testausta. Tilaaja vaati projektin alusta lähtien esikäyttäjien käyttämistä, mutta toisaalta esimerkiksi järjestelmätestaus suunniteltiin täysin projektisuunnitelman [7] keskeyttämisen jälkeen. Luvussa kuvataan sovelluksen laadun varmistamiseen käytettyjä tapoja.

7.1 Lähdekoodin käytänteet

Lähdekoodin käytänteistä sovittiin ensimmäisessä palaverissa [13], että projektissa käytetyn alustan suosituksia ja käytänteitä noudatetaan ohjelmoimissa. Hyvää ohjelmointityyliä tavoitellessaan ryhmän oli siis sekä seurattava Ruby-ohjelmointikielen [2] että Ruby on Rails -kehiksen suosituksia, minkä lisäksi Gitorious-projekti määritteli joitain omia ohjelmointikäytänteitä [5].

Ruby-kielen osalta hyvää ohjelmointityyliä noudattava luokka- ja metodimäärittelyä havainnollistaa seuraava:

```
class Repository < ActiveRecord::Base
  include ActiveMessaging::MessageSender
  ...

  def can_view?(a_user)
    if REPO_VIEWABLE_EVERYONE
      return true
    else
      return viewer?(a_user)
    end
  end
end
...

end
```

Suunniteltujen käytänteiden noudattaminen toteutui hyvin, mitä auttoi oppiva ja keskusteleva työympäristö. Lisäksi auttoivat koodikatselmoinnit, joissa teknisen oh-

jaajan Antti-Juhani Kaijanahon johdolla käsiteltiin myös ohjelmointityyliin liittyviä asioita.

Kehityksessä käytettyjä käytänteitä kuvataan tarkemmin sovellusraportissa [16].

7.2 Pariohjelmointi

Verso-projektissa käytettiin ohjelmoinnin yhteydessä välillä pariohjelmointia. Pariohjelmointia varten ei kuitenkaan kehitetty tai tarvittu selkeitä käytänteitä tai suunnitelmaa, sillä projektissa oli paikalla yhtäaikaisesti korkeintaan kolme kehittäjää projektipäällikön lisäksi. Kolmen kehittäjän osalta etenkin edeltävän palaverin sihteerin aikaa kului pöytäkirjan laatimiseen, joten toisinaan neljästäänkin työskennellessä kehitystä teki projektin jäsenistä korkeintaan kaksi.

Käytännössä projektissa ohjelmoitiin parityönä vaativat ja paljon tutkimista vaativat kokonaisuudet. Parin käyttö haastavissa tilanteissa vaikutti auttavan ratkaisutapojen hahmottelemisessa huomattavasti. Lisäksi se pitkälti esti turhauttavista ohjelmointiosuuksista yksin kärsimisen, vähensi virheiden jäämistä ohjelmakoodiin ja levitti automaattisesti tietoutta ohjelmakoodiin tehtävistä muutoksista muulle ryhmälle.

Vastavuoroisesti triviaalimmat kehitystehtävät toteutettiin usein yksin ohjelmoiden, mikä toisaalta mahdollisti joidenkin ominaisuuksien kehityksen henkilöitymisen projektissa. Henkilöitymisestä ei kuitenkaan muodostunut ongelmaa, koska projektiryhmä näki pääsääntöisesti toisiaan monen tunnin ajan ja tieto vaihtui projektihuoneessa helposti.

7.3 Integraatiotestaukset

Sovelluksen versiohallinnassa keskeneräinen lähdekoodi pidettiin erillään varsinaisesta päähaaran (engl. *master branch*) ohjelmakoodista, kunnes kehittäjä totesi sen olevan valmis esikäytettäväksi. Versiohallinnan käytänteitä on kuvattu tarkemmin luvussa 8.

Käytännössä toteuttaessaan ominaisuutta sekä sen toteuttamisen jälkeen kehittäjä varmisti toteuttamansa ominaisuuden toimimisen omalla kehityspalvelimellaan osana sovellusta ennen sen siirtämistä osaksi varsinaista sovelluskoodia. Kehittäjä

testasi kehittämänsä toiminnon lisäksi sovelluksen muista osioista ne, joihin kehittäjä arveli ominaisuuden aiheuttaman muutoksen vaikuttavan. Ominaisuutta käyttöönotettaessa ei tehty raskasta järjestelmätestausta, vaan kehittäjän oman testauksen jälkeen hän siirsi koodin projektiorganisaation ja esikäyttäjien käyttöön.

7.4 Automaattiset poikkeustiedotteet

Projektissa ei alunperin suunniteltu käytettävän automaattisia poikkeustiedotteita, vaan ne olivat myöhemmin löydetty ja käyttöön otettu Ruby on Rails -alustan tarjoama toiminto.

Gitorious-sovellus mahdollisti poikkeustilanteiden automaattisen raportoinnin. Käytännössä aina, kun esikäytössä olleessa sovelluksessa tapahtui poikkeus, lähti siitä sähköpostiviesti koko projektiryhmälle. Näin sovellukseen jäämien virheiden löytäminen ei ulkopuolistenkaan esikäyttäjien puolesta nojautunut ilmoitusahkeruuteen, vaan virhetilanteesta ja siihen johtaneesta sivupyynnöstä saatiin aina tieto.

Poikkeuksen tapahduttua ryhmä päätti itsenäisesti, miten tilanteeseen oli tarpeen reagoida. Sovellus tuottaa lisäksi viestejä kymmenkuntaan eri lokitiedostoon, josta oli usein löydettävissä lisää tietoa virhetilanteen toistamiseksi kehitystä varten.

7.5 Esikäyttäjien hyödyntäminen

Projektin aikana sovellusta kutsuttiin käyttämään parikymmentä esikäyttäjää, joilta pyydettiin palautetta sovelluksen ominaisuuksista ja käytettävyydestä. Esikäyttäjien hyödyntämisestä laadittiin erillinen suunnitelma [9]. Esikäyttäjien hyödyntäminen vastasi suunniteltua.

Tilaaaja halusi saada projektiorganisaation ulkopuolisia jäseniä esikäyttäjiksi mahdollisimman aikaisessa vaiheessa. Esikäytön toteuttamisesta laaditun suunnitelman valmistuminen kuitenkin venyi siihen esitettyjen parannusehdotusten vuoksi usean projektipalaverin ajan. Yli kuukausi projektiryhmän oman käytön aloittamisen jälkeen suunnitelma hyväksyttiin ja YouSource-sovellus tarjottiin esikäytettäväksi 30.3.2010.

Palvelimen `versotest.it.jyu.fi` verkkokonfiguraatio salli aluksi pääsyn vain tiettyihin Jyväskylän yliopiston tietotekniikan laitoksen projekti- ja henkilökunnan

tiloista. Tämä esti käytännössä täysin opiskelijoiden hyödyntämisen esikäytössä. Sovellus avattiin koko Jyväskylän yliopiston verkkoon 13.4.2010, jolloin projektiorganisaation oman käytön alkamisesta oli kulunut noin 7 viikkoa. Avaus mahdollisti viimein opiskelijoille ja kotoaan esikäyttävälle tutkijoille sovelluksen käytön.

Projektiorganisaation ulkopuolinen esikäyttö jäi Verso-projektin osalta vähäiseksi. Sovellukseen kirjautui projektin aikana noin 30 projektiorganisaation ulkopuolista käyttäjää, joista vain harva loi sovellukseen omia tietovarastoja. Myös esikäyttäjien viestintään varattu sähköpostilista `yousource-users.group@korppi.jyu.fi` jäi lähes täysin yksipuoliseksi projektiryhmän tiedotuskanavaksi.

On hankala sanoa, olisivatko aiempi verkosta pääsyn helpottaminen ja 19.4.2010 julkistetun Korppi-tunnuksilla tapahtuvan kirjautumisen julkistaminen aiemmin vaikuttaneet ratkaisevasti esikäyttöön. Projektin aikana `versotest.it.jyu.fi`-palvelimelle ei kuitenkaan päässyt Jyväskylän yliopiston verkon ulkopuolelta, joten varsinaista omien lähdekoodien julkistamista yliopistoverkon ulkopuolelle esikäyttäjä ei projektin aikana voinut tehdä.

Projektin jälkeisen jatkokehityksen jälkeen palvelu avattiin tuotantokäyttöön syyskuussa uudelle palvelimelle `yousource.it.jyu.fi`, joka mahdollistaa julkistamisen yliopistoverkon ulkopuolellekin.

7.6 Käytettävyyispäivä

Meeri Mäntylä piti projektiryhmälle käytettävyyслуennon 29.3.2010, jossa käsiteltiin sovelluksen käytettävyyteen liittyviä kriteerejä. Lisäksi ryhmä toteutti ryhmätyönä WWW-sivuston käytettävyytestauksen, mikä toimi hyvänä harjoituksena myöhemmin toteutetuille käytettävyytestauksille. Käytettävyyispäivä toteutui suunnitellusti.

Luennon lisäksi Mäntylä kävi läpi YouSource-sovellusta ja nosti esiin havaitsemiaan käytettävyyden seikkoja. Selkeitä käytettävyysspuutteita kirjattiin tiketeiksi, minkä lisäksi käytettävyyispäivästä laadittiin muistio [22].

7.7 Käytettävyytestaukset

Verso-projekti suoritti YouSource-sovelluksen käytön aloittamiseen liittyvistä toimista kaksi käytettävyytestausta, joissa käytettiin tilaajan nimeämiä testihenkilöitä. Käytettävyytestauksissa esille nousseet ongelmat ja huomiot kirjattiin muistioiksi ja tiketeiksi, minkä lisäksi niiden yleiset huomiot käytiin läpi projektipalaverissa. Esikäyttäjien hyödyntämissuunnitelma [9] sekä ensimmäisestä ja toisesta käytettävyytestauksesta laaditut muistiot [12] ja [14] löytyvät projektin dokumentaatiosta.

7.8 Koodikatselmoinnit

Sovellusprojekti-kurssiin kuuluu luonnostaan sovelluksen koodikatselmoiteja. Sovitut kaksi koodikatselmointia toteutuivat suunnitellusti.

Projekti asetti vieraan kielen Rubyn ja sovelluskehyksensä Ruby on Railsin osalta haasteita sekä projektiryhmälle että tekniselle ohjaajalle, koska molemmille osapuolille nämä olivat entuudestaan vieraita. Projektin tekninen ohjaaja Antti-Juhani Kaijanaho oli projektia varten tutustunut Rubyyn ja Ruby on Railsiin, joten hän pystyi koodikatselmoinneissa nostamaan huomioissaan esille myös monia kielen ja sovelluskehysten käytänteisiin liittyviä seikkoja.

Projekti käytti YouSource-sovelluksen pohjana valmista Gitorious-järjestelmää, joten kaikkea projektin toteuttamaa lähdekoodia ei katselmoinneissa voitu käydä läpi. Ryhmä valitsi katselmoinneissa läpikäytävät ohjelmaosiot niiden tärkeyden ja valmiusasteen perusteella.

Sekä ensimmäisestä että toisesta katselmoinnista ryhmä laati muistiot [6] ja [17]. Kaijanaho lähetti lisäksi 2. katselmoinnin jälkeen lähdekoodista huomionsa [1] projektin sähköpostilistalle.

7.9 Järjestelmätestaukset

Kuten luvun johdannossa mainittiin, järjestelmätestaus suunniteltiin ja toteutettiin täysin projektisuunnitelman [7] laatimisen keskeyttämisen jälkeen, eikä järjestelmätestausta siis suunniteltu projektin alussa suoritettavan.

Tero Hänninen laati YouSource-sovelluksen järjestelmätestaussuunnitelman [23], joka määrittelee järjestelmän toimintojen hyväksymiskriteerit kohdittain.

Ensimmäisessä sovelluksen RC1-version järjestelmätestauksessa [24] löydettiin kaksi projektissa toteutettuun näkyvyyksien säätelyyn liittyvää virhettä ja SVN-peilaukseen liittyvä virhe. Järjestelmätestauksessa havaitut näkyvyyksiin liittyvät ohjelmointivirheet korjattiin toiseen testikertaan [25], mutta palvelinympäristöistä johtuvat SVN-peilaukseen liittyvät ongelmat palvelinympäristössä sovittiin tilaajan kanssa jatkokehityksessä ratkaistaviksi.

8 Versiohallinnan käyttö

Versiohallinta oli monella tavalla tärkeä osa YouSource-sovelluksen kehitystä, Verso-projektin käytänteitä ja oppimiskokemusta. Luvussa kuvataan suunniteltua ja toteutunutta versiohallinnan käyttöä projektissa.

Kesken jääneessä projektisuunnitelmassa [7] kuvattiin projektissa käytettävät versiohallintakäytännöt, jotka vastasivat toteutuneita niiltä osin kuin niitä oli määritetty. Kuitenkin, kuten projektiryhmä myöhemmin huomasi tarjotessaan ominaisuuksia muille kehittäjille, suunnitelma olisi voinut olla monella tavalla parempi.

Toteutuneet tavat käyttää versiohallintaa toimivat kuitenkin Verso-projektin ajan hienosti. Koska versiohallinnan käyttö oli valtaosalle projektiryhmästä vierasta, oppimisen kannalta Verso-projekti oli erittäin hyödyllinen. Myös havaittujen ongelmien korjaaminen on mahdollista, ja ne pakottavat opettelemaan edelleen uusia menetelmiä.

8.1 Suunniteltu haarojenkäyttöstrategia

Projektin tulokset tallennettiin **Git-versiohallintaan** kahteen eri tietovarastoon. Sovellusta kehitettiin omaan tietovarastoonsa sekä omaan tietovarastoonsa projektin dokumentteja, kuten muistioita, palaverien pöytäkirjoja ja väliesittelymateriaaleja. Väliaikaishaaroja (engl. *topic branch*) käytettiin sekä projektitietovarastossa että etenkin sovelluksen tietovarastossa. Väliaikaishaara sisälsi esimerkiksi tiettyyn ominaisuuteen tai asiakokonaisuuteen (kuten tiettyyn pöytäkirjaan) liittyvät päivitykset.

Sovelluksen kehityksen kannalta **haarojen käyttäminen** oli tärkeää, koska sovelluksen päähaara oli julkistamishaara esikäyttöä varten. Epävakaat ohjelmakoodit oli pidettävä omissa haaroissaan niiden koekäyttöön valmistumiseen asti. Kun kehitetty sovelluksen ominaisuus oli varmistettu integraatiotestauksella kehittäjän omalla kehityspalvelimella, sen haara yhdistettiin (engl. *merge*) päähaaraan (engl. *master branch*), ja ominaisuuden historian sisältävä haara jätettiin versiohallintaan. Ominaisuuksien tarjoamisen Gitorious-sovellukselle ajateltiin olevan helpompaa, kun projektissa toteutettavien ominaisuuksien haarat olisivat lähdekoodeja tarjotessa tallessa ja erillään muusta ohjelmakoodista.

Projektin ajan käytetty **haarojenkäyttöstrategia** (engl. *branching strategy*) toimi projektin oman kehityksen tarpeisiin hienosti. Vasta toukokuussa tehtäessä yhdistämis-

pyyntöjä (engl. *merge request*) projektiryhmä huomasi haarojenkäyttöstrategiassaan luvussa 8.2 kuvattavia puutteita, jotka tekivät muokkauspyyntöjen lähettämisestä alkuperäiseen Gitorious-sovellukseen haastavaa.

8.2 Havaitut puutteet versiohallinnan käytössä

Vaikka käytetty menetelmä suurempien ominaisuuksien pitämisestä erillään päähaarasta toimi projektin ajan, projektiryhmä huomasi projektin päättyessä, että haarojen kanssa olisi pitänyt olla huomattavasti tarkempi.

YouSource-sovelluksen **kehitys tehtiin Gitorious-sovelluksen kantaversion** (engl. *mainline*) **päähaaran päälle** lisäten siihen ryhmän omia muutoksia ja yhdistäen (engl. *merge*) siihen kantaversion päähaarassa tapahtuneet muutokset. Verso-projektin kehittämät ominaisuudet kehitettiin tämän yhdistetyn päähaaran päälle.

Kuitenkin, ominaisuuksien tarjoamista varten olisi ollut parempi pohjata ryhmän kehitys **täysin koskemattoman Gitorious-sovelluksen päähaaran päälle**. Näin omat muutokset, ominaisuudet ja korjaukset olisivat olleet suoraan yhteensopivia alkuperäisen Gitorious-kantaversion ja monien sen kloonien kanssa. YouSource-sovelluksen logo, omat tekstit ja pienet käyttöliittymämuutokset olisi lisäksi pitänyt pitää muiden ominaisuuksien tavoin omassa haarassaan, jota olisi pidetty yhteensopivana Gitorious-kantaversion päähaaran kanssa.

Projektin alussa **yhdistämispyyntöjen tekemiseen ei perehdytty käytännössä**. Perehtymisen kautta olisi selvinnyt, että itse asiassa jokainen yhdistämispyynnön arvoinen muokkaus tai korjaus kannattaisi pitää omassa haarassaan. Näin ollen Verso-projektin aikana tarvittujen haarojen määrä olisi kasvanut merkittävästi nykyisestä noin kymmenestä moniin kymmeneen. Sopivalla nimeämiskäytännöllä tämä ei kuitenkaan olisi ollut ongelma. Haarojen tekeminen ei varsinaisesti ole yleinen vaatimus, mutta Gitorious-sovelluksessa yhdistämispyyntöjen tekeminen olisi siten helppoa, koska yhdistämispyyntötoiminto ei mahdollista vain tiettyjen muokkauksien (engl. *commit*) poimimista (engl. *cherry pick*).

Yleinen ongelma versiohallinnan käytössä ovat **toisistaan riippuvat ominaisuudet**, joiden toteuttamisen osalta tilanne on samankaltainen kuin niiden työajankirjauksissakin. Ohjelmoidessa toteutettu toiminnallisuus on mahdollisesti vaatinut jonkin ongelman korjaamista tai toisen toiminnallisuuden kehittämistä ennen työn alla olleen toiminnallisuuden toteuttamista. Kun työn alla ollut toiminnallisuus valmis-

tuu, siihen liittyvät muokkaukset pitäisi lähettää versiohallintaan vasta toimintoa varten kehitettyjen ominaisuuksien jälkeen.

Selkeiden muutoksien lähettämisestä on joissain tilanteissa työlästä pitää kiinni. Riskinä kuitenkin on, että versiohistoriaan päätyy **useita toimintoja sisältäviä committeja**. Työn alla ollut ominaisuus on mahdollisesti vaatinut korjauksia muissa haaroissa kehitettyihin toimintoihin, ja muutoksien lähettäminen oikein vaatisi niiden työlästä synkronointia. Kuitenkin, muutoksien jakamiseen käytetty työ niitä lähettäessä on kevyempi kuin muutoksien jakaminen myöhemmin osiin.

Muokkausviesteihin (engl. *commit message*) olisi ollut hyödyllistä kirjoittaa viestin alkuun työn alla olleen Trac-tiketin numero. Tämä olisi ensinnäkin auttanut You-Source-tietovaraston jälkikäteen tehtyä rakenteen muokkausta, jota on kuvattu tarkemmin luvussa 8.4. Toiseksi, käyttäen tikettinumeron lisäämisestä olisi osaltaan auttanut varmistamaan, että muokattavasta asiasta on tehty Trac-tiketti. Tämä käytäntö ei kuitenkaan tullut projektiryhmälle ennen toukokuuta mieleen. Lisäksi projektihallintasovelluksen käyttämiseen ei sovellusprojekteissa ole valmiita käytänteitä, koska näin niiden käyttö projekteissa on toistaiseksi vapaaehtoista ja harvinaista.

Ryhmä teki myös muutaman **selkeän virheen**. Esimerkiksi, kun tietyn toiminnon kehityshaaran kehittämistä varten tarvittiin päähaarassa jokin tuore muutos (joka ei siis automaattisesti näy kehityshaarassa), muutos otettiin kehityshaaraan yhdistämällä siihen päähaaraan tehdyt muutokset. Tilanne olisi **ehdottomasti** pitänyt ratkaista siirtämällä kehityshaara alkamaan uudesta alkupisteestä (engl. *rebase*), jotta kehityshaaraan ei päätyisi ylimääräisiä muutoksia.

8.3 Soveltuvampi haarojenkäyttöstrategia

Idea soveltuvasta haarojenkäyttöstrategiasta syntyi toukokuussa. Kuten luvussa 8.2 kuvattiin, Gitorious-kantasovellus olisi pitänyt pitää erillään kaikesta omasta kehityksestä sekä kehittää toiminnallisuudet ja korjaukset kantasovelluksen päälle. Haaroja tulee mahdollisesti paljon, mutta sopivilla nimeämiskäytännöillä ne pysyvät hallittavissa. Vaikkapa nimeämällä ne *fix/*- ja *feature/* -alkuisesti sekä mahdollisesti lisäämällä niihin Trac-tiketin numeron pääsisi nimeämiskäytännöissä jo pitkälle. Omat sovelluksen personoinnit (engl. *customization*) olisi pitänyt pitää ominaisuuksien tavoin omassa haarassaan.

Tärkeintä olisi **koota erillisistä haaroista koostuva oma sovellus omaan haaraan**

käyttöä varten. Nykyisen "YouSource-tietovaraston" sijaan olisi parempi olla YouSource vain yhtenä haarana Gitorious-kantasovellukseen pohjautuvassa tietovarastossa. Haara voi yhtä hyvin olla palvelimen nimi etenkin, jos personointeja halutaan jatkossa erilaisia.

Myös edellä kuvattu haarojenkäyttöstrategia **asettaa haasteita**. Kehityksen tapahtuessa jatkuvasti ylläpidettävissä haaroissa, niitä pitää toisinaan esimerkiksi synkronoida alkavaksi uudemmasta pisteestä Gitorious-kantasovelluksen päähaarasta. Tämän tekeminen vaatii sovitut ja yhtenäiset käytänteet kehittäjien keskuudessa, sillä hallitsematon kehityshaaran alkupisteen siirtäminen voi johtaa vaikeuksiin muiden kehittäjien synkronoidessa tietovarastoaan.

8.4 Tietovaraston rakenteen jatkokehitys

Koska YouSource-sovelluksen kannalta on hyödyllistä tarjota ominaisuuksista merge requesteja Gitorious-kehittäjille, ominaisuuksien "perkaamista" Gitorious-kantasovelluksen päähaaran päälle rakentuvaksi kannattaa jatkaa. Verso-projektin aikana sitä ehdittiin neljän korjauksen osalta jo tekemään, joiden pohjalta tehdyistä muutospyyntöistä kolme hyväksyttiin projektin aikana osaksi Gitorious-kantasovellusta.

Haaroja muokattaessa on tärkeää, että **muokkauksien kuvauksissa** kerrotaan selkeästi, mitä toiminnallisuutta muokkaus koskee. Projektiryhmä on projektin aikana valvonut omatoimisesti muokkausviestien sisältöä ja kuvaavuutta, mikä auttaa muun muassa tietovaraston rakennetta ja versiohistoriaa jälkeenpäin muokatessa suuresti.

9 Trac-käytänteet

Projektissa käytettiin vaatimusmäärittelyn hallintaan Trac-ohjelmistoa. Koska erillistä vikojenseurantaohjelmistoa (engl. *bugtracker*) ei käytetty, projektiryhmän ja tilaajan välisen tehtävähallinnan lisäksi Traciin kirjattiin myös käyttäjien kautta tulleet vikailmoitukset ja toiveet.

Trac-sovellusta käytettiin pääosin projektisuunnitelman [7] kuvaamalla tavalla, mutta Trac osoittautui myös joiltain osin puutteelliseksi. Projektin aikana havaitut puutteet olivat pääasiassa käytettävyysongelmia.

9.1 Käyttöoikeudet

Trac-järjestelmän käyttöoikeudet vastasivat suunniteltua. Verso-projektin Tracin selaamiseen ei vaadittu kirjautumista, mutta muokkaaminen oli mahdollista vain luvussa 4.1 mainituille projektiorganisaation jäsenille.

Avoin pääsy antoi mahdollisuuden selata tehtäviä myös esikäyttäjille ja muille projektista kiinnostuneille. Projektiorganisaation ulkopuolisen ilmoittajan kanssa yhteyttä pitäessä voitiin viitata suoraan Tracin tiketteihin.

9.2 Tikettien muutoksista tiedottaminen

Trac-tiketteihin merkittiin muutosilmoitusten vastaanottajaksi cc-kenttään projektiorganisaation sähköpostilista. Lisäksi, jos ilmoitus tuli projektiorganisaation ulkopuolelta (esimerkiksi esikäyttäjiltä), jakelulistaan lisättiin ilmoittajan sähköpostiosoite, jotta ilmoittaja pysyi ajantasalla tehtävään liittyvistä muutoksista.

Muutosilmoitusten lähettäminen projektiorganisaation postilistalle nähtiin tarpeelliseksi, jotta tilaaja, projektiryhmä ja ohjaajat saavat automaattisesti tiedon muutoksista. On vaikea sanoa, olivatko sähköposti-ilmoitukset lopulta hyödyllisiä. Tikettien muutoksien tekeminen oli pitkälle käsityötä parempien toimintojen puuttuessa, ja kustakin muutoksesta lähti erikseen ilmoitus, vaikkei se tiketin sisältöön vaikuttanutkaan. Jos projektin päättyessä haluaisi esimerkiksi vaihtaa versionumeron kaikkiin tiketteihin (jonka siis joutuisi tekemään käsin kaikille tiketeille erikseen) lähtisi tästä toimenpiteestä lähes **150 sähköpostia ilman viestinnällistä hyötyä**.

9.3 Tiketteihin kirjatut tiedot

Tracia ei käytetty kaikkien Verso-projektin tehtävien hallintaan, vaan ainoastaan siinä kehitetyn YouSource-sovelluksen ja sen esikäyttöä varten pystytetyn testipalvelimen tehtävien osalta. Pääsääntöisesti tehtävä tai idea tuli kirjata Traciin, mikäli se liittyi sovellukseen. Kuten muissakin sovellukseen liittyvissä dokumenteissa, Tracissa käytettiin kirjauskielenä englantia.

Tiketin **tiivistelmäksi** (engl. *summary*) tiketin lisääjä kirjasi mahdollisimman yksiselitteisen kuvauksen, koska tiivistelmä toimi tiketin otsikkona. Selkeän otsikon avulla tiketti löytyi tikettilistoista ja sähköpostikansioista helpommin.

Tiketin **kuvausta** (engl. *description*) käytettiin selvittämään tarkemmin tiketissä käsiteltävä asia. Kuvaukseen pystyi kirjaamaan esimerkiksi tikettiin liittyviä kehitysratkaisuja tai sovellukseen lisättäviä tekstejä.

Tiketin luominen oli samalla pienimuotoinen keskustelunavaus tiketissä kuvattuun tehtävään liittyen. Tiketin **kommenteissa** (engl. *comment*) oli mahdollista kysyä tehtävään liittyen tarkennuksia, joiden perusteella sen lisääjä pystyi esimerkiksi muokkaamaan tiketin kuvausta tarkemmaksi.

Tikettien **otsikot eivät olleet muuttumattomia**. Mikäli tiketin otsikko ei kuvannut siinä käsiteltäviä asioita kyllin selkeästi, kaikilla projektiorganisaatioon kuuluvilla oli mahdollisuus muokata sitä paremmaksi. Tikettien vanha otsikko jäi tikettiin talteen, joten vaatimukseen liittyvällä informaatiolla ei ollut mahdollisuutta hukkoa muutoksissa.

Otsikkoa muokatessa on vaarana, ettei otsikko enää muokkauksen jälkeen kuvannut tilaajan hyväksymää asiaa. Tämä riski ei kuitenkaan ollut projektissa todellinen, koska tiketit toimivat pääosin muistin tukena palaverissa käsitellyistä asioista, ja ne tarkastettiin ennen toteutusvaiheiden aloittamista palaverissa.

9.4 Tikettityypit

Projektisuunnitelmassa [7] kuvatut tikettityypit osoittautuivat käyttökelpoisiksi, ja projekti toteutui pitkälti niitä käyttäen. Trac-sovellus asetti kuitenkin joidenkin puutteidensa osalta haasteita tikettien ylläpitoon, kuten luvussa 9.2 kuvattiin.

Taulukossa 9.1 on kuvattu erilaiset Tracissa käytetyt tikettityypit. Taulukon termillä omistaja tarkoitetaan sitä tahoa, jonka kommunikointivälineeksi tikettityyppi oli tarkoitettu.

Tikettityyppi	Omistaja	Kuvaus
Feature	Tilaaja	Sovellukseen hyväksytty ominaisuus.
Chore	Tilaaja	Projektiryhmän muu tehtävä.
Wish	Kuka tahansa	Kehitysidea tai muu toive.
Bug	Kuka tahansa	Sovelluksesta löytynyt vika.
task	Projektiryhmä	Pieni kehitystehtävä.

Taulukko 9.1: Trac-tikettien tyypit.

Suunnitellusta tikettien käytöstä poikettiin lähinnä task-tyyppisten tehtävien kohdalla. Tikettityyppi task oli kirjoitettu muista poiketen siksi, että se oli suunniteltu täysin projektiryhmän sisäiseen käyttöön sovelluksen kehittämistehtävien tueksi. Ajatus oli käyttää sitä laajempien tehtävien pilkkomisessa osiin, mutta useiden tikettitasojen mallintaminen Tracissa koettiin niin hankalaksi, että ryhmä ei lopulta käyttänyt task-tyyppisiä tikettejä, vaan kommentoi tikettejä tarvittaessa lisätiedon välittämiseksi.

Verso-projektin Traciin oli järjestetty lukuoikeus kaikille, mutta tietojen muokkaamiseen olivat oikeutettuja vain projektiorganisaatioon kuuluvat käyttäjät. Kehitysideoita ja bugeja tuli kuitenkin myös projektiorganisaation ulkopuolelta. Omistajalla "Kuka tahansa" tarkoitetaan kenen tahansa lähettämää huomiota, jonka vastaanottaja projektiorganisaatiossa teki siitä tiketin.

9.5 Prioriteetit

Projektsuunnitelmassa [7] kuvatut tikettien prioriteetit olivat kaikki käytössä, ja projekti toteutui ilman muutoksia niihin.

Prioriteetit (engl. *priority*) suunniteltiin käytettäväksi Feature-tyyppisten tikettien kanssa sekä muiden tikettityyppien kanssa niitä käytettiin suuntaa-antavana lisätietona. Prioriteetti ei suoraan määrättänyt ominaisuuden kehitetyksi tulemisesta Verso-projektissa, vaan niitä hyödynnettiin valittaessa ja kiinnittäessä tehtäviä toteutusvaiheisiin.

Prioriteetti	Kuvaus
Mandatory	Pakolliseksi nähtävä, toteutetaan ensimmäisten joukossa.
Important	Tärkeä, ei pakollinen, toteutetaan ajan salliessa.
Useful	Hyödyllinen, ei käytön kannalta tärkeä, toteutetaan ajan salliessa.
Left out	Projektin osalta tarpeettomaksi nähty.
–	Toistaiseksi priorisoimaton ominaisuus.

Taulukko 9.2: Trac-tikettien prioriteetit selityksineen.

10 Tehtävien työmäärät ja työnjako

Luvussa tarkastellaan toteutunutta projektin vastuualueiden ja tehtävien jakautumista sekä verrataan niitä suunniteltuun. Dokumentoinnin ja ohjelmoinnin vastuualueisiin ei tullut muutoksia projektin alussa suunniteltuihin.

10.1 Tehtävien jakautuminen

Taulukossa 10.1 on esitetty suunnitellut ja toteutuneet työmäärät sekä niiden jakautuminen ryhmän jäsenten kesken. Taulukosta ilmeisin havainto ovat suurelta osin puuttuvat suunnitellut tunnit, mikä johtuu myöhästyneestä ja kesken jääneestä projektisuunnitelmasta.

Toteutuman ja suunnitelman **vertaaminen projektin päätteeksi** ei ole mielekäästä kuin osittain. Koko projektin tuntien hallintaan kiinnitettiin kunnolla huomiota vasta projektin loppupuolella.

Yleisesti projektiin käytettiin arvioitua enemmän tunteja. Projektipäällikkö esitti projektin toisessa palaverissa [15] projektin tuntiarvioksi kultakin jäseneltä 250 tuntia, mitä pidettiin palaverissa heti matalana arviona. Projektin tuntien suunnittelu jäi puutteelliseksi, mihin suurin syy oli projektipäällikön projektin alussa liian pieni panostus projektisuunnitelmaa kohtaan.

Kohtiin **esitutkimus, projektin hallinta, suunnittelu, toteutus, testaus ja viimeistely** ei tunteja oltu suunniteltu laisinkaan. Suurelta osin arvioinnin puuttuminen suunnitelmassa johtuu projektin tavasta käyttää määrämittäisiä jaksoja, joiden sisältö kiinnitetään vasta jakson alussa. Vaikka osa toteutetuista kokonaisuuksista oli ajateltu tehtäviksi yhden jakson aikana, vasta viimeisessä jaksossa tehtävien laajuuksia arvioitiin tuntien tarkkuudella, jotta viimeiseen jaksoon valitut asiat voitaisiin luvata tehdä. Yhdessä edellä mainitut tehtäväkokonaisuudet muodostavat yli puolet toteutuneista tunneista.

Esikäyttäjät-kohdan toteutuneet tunnit (24) jäivät paljon suunnitellusta (62), mikä johtui lähinnä odotettua vähäisemmästä esikäytöstä. Muutoin esikäyttöön suunnitellut tunnit toteutuivat odotetunkaltaisesti. Heikki vastasi esikäyttäjien tiedottamisesta, mihin kului suurin osa esikäyttöön vaaditusta ajasta. Loput tunnit kuluivat lähinnä muun ryhmän suorittamiin käytettävyytestestauksiin ja niiden dokumentointiin.

Tehtävät Vaihe	Juho		Marko		Tero		Heikki		Yhteensä	
	S	T	S	T	S	T	S	T	S	T
Esikäyttäjät	10	7	10	1	10	2	32	14	62	24
esikäyttötiedotus	0	0	0	0	0	0	8	1	8	1
esikäyttöviestintä	6	0	6	0	6	0	14	3	32	3
käytettävyydestaus	2	7	2	1	2	2	0	0	6	10
haastattelu	2	0	2	0	2	0	0	0	6	0
raportointi	0	0	0	0	0	0	0	0	0	0
suunnittelu	0	0	0	0	0	0	10	10	10	10
Esitutkimus	-	28	-	55	-	68	-	14	-	165
tutustuminen	-	28	-	55	-	68	-	14	-	165
Oheiskurssi	50	93	50	84	50	64	50	116	200	357
dokumenttien kirjoitusasu	-	28	-	6	-	8	-	52	-	94
esittelyt	15	30	15	27	15	22	15	20	60	99
katselmoinnit	5	3	5	10	5	3	5	6	20	22
koulutus	30	27	30	36	30	28	30	33	120	124
raportointi	-	5	-	5	-	3	-	5	-	18
Palaverit	20	45	35	50	35	34	35	57	125	186
ldap-autentikointi	-	-	-	-	-	-	-	1	-	1
palaveri	15	19	15	23	15	15	15	46	60	103
raportointi	20	26	20	27	20	19	20	10	80	82
Projektin hallinta	0	1	0	4	0	17	0	202	0	224
muut tehtävät	0	0	0	3	0	16	0	5	0	24
raportointi	0	0	0	0	0	0	0	68	0	68
suunnittelu	0	0	0	0	0	0	0	93	0	93
tiedotus	0	1	0	1	0	1	0	36	0	39
Sovellusraportti	50	108	5	0	5	0	5	0	65	108
sovellusraportti	50	108	5	0	5	0	5	0	65	108
Suunnittelu	-	32	-	11	-	4	-	18	-	65
suunnittelu	-	32	-	11	-	4	-	18	-	65
Testaus	-	2	-	2	-	32	-	2	-	38
järjestelmättestaus	-	0	-	2	-	6	-	2	-	10
raportointi	-	0	-	0	-	4	-	0	-	4
suunnittelut	-	2	-	0	-	22	-	0	-	24
Toteutus	-	192	-	234	-	200	-	73	-	699
käyttöliittymä	-	93	-	8	-	19	-	4	-	124
ldap-autentikointi	-	23	-	0	-	0	-	18	-	41
metatiedostot	-	15	-	34	-	74	-	0	-	123
ohjeet ja neuvonta	-	5	-	36	-	0	-	0	-	41
oikeudet ja näkyvydet	-	5	-	99	-	25	-	0	-	129
päivitystavat	-	40	-	26	-	74	-	0	-	140
tukitehtävät	-	11	-	31	-	8	-	51	-	101
Viimeistely	-	4	-	42	-	62	-	21	-	129
käyttöliittymä	-	0	-	6	-	2	-	0	-	8
merge requestit	-	0	-	0	-	6	-	0	-	6
lähdekoodin siistiminen	-	4	-	20	-	44	-	13	-	81
suunnittelu	-	0	-	6	-	0	-	7	-	13
tulosten luovutus	-	0	-	10	-	10	-	1	-	21
Yhteensä	130	512	100	483	100	483	122	517	452	1995

Kuva 10.1: Suunnitellut ja toteutuneet työtunnit.

Esitutkimus-kohtaan on kirjattu tunteja lähinnä projektin alussa. Esitutkimus oli toteutusta edeltävää pohjatyötä ohjelmointiympäristöjen tai Git-versiohallinnan opetteluun. Esitutkimukselle ei varattu tiettyä tuntikiintiötä, ja sitä teki projektin alussa koko ryhmä. Teron tunteihin (68) kuului nimenomaan Git-versiohallinnan opettelu, jonka seurauksena Terolla oli siitä projektin aikana parhaat tiedot.

Oheiskurssiin on merkitty opintojakson *Sovellusprojektin viestintää, hallintaa ja työkaluja* sekä kirjoitus- ja puheviestinnän kurssin *Projektiviestintä IT-alalla* edellyttämiä työtehtäviä. Erityisesti niitä olivat projektin aikana luennot sekä dokumenttien kieliasujen korjaaminen. Projektin väli- ja loppuesittelyihin käytetty valmistautumisaika on myös kirjattu oheiskurssin alle.

Palaverihin on kirjattu palaverihin kulunut aika projektin jäseniltä sisältäen palaverieita edeltävän valmistelun sekä pöytäkirjojen laatimisen. Heikki laati palaverihin tilakatsauksen, minkä muodostamiseen kulunut aika kirjattiin pääosin *Projektin hallinta* -kokonaisuuteen raportoinnin alle, mutta se näkyy myös palaverihin valmistautumisena.

Projektin hallintaan käytettiin projektissa yhteensä 224 tuntia, johon kirjatusta tunteista suurin osa muodostuu projektisuunnitelman ja -raportin laatimisesta. Heikki vastasi projektipäällikkönä projektin hallinnasta lähes täysin. Projektisuunnitelmaan on kuitenkin merkitty myös projektisuunnitelman kieliasun korjaamiseen käytettyjä tunteja, jotka kirjauskäytänteiden päivittyessä on muissa dokumenteissa kohdennettu **Oheiskurssi**-tehtäväkokonaisuuden alle.

Juhon laatima **sovellusraportti** suunniteltiin 50 tunnin laajuiseksi Juholle ja 5 tunnin laajuiseksi muille. Muille suunniteltujen ja merkittyjen tuntien osuus on vähäinen, koska raportin edellyttämä yhteinen sovelluksen toiminnan selvittäminen ei näyttänyt vaativan tai vaatinut erikseen merkittäviä rupeamia. Juhon sovellusraporttiin käyttämä aika (108 tuntia) oli merkittävästi suunniteltua suurempi.

Kohdan **suunnittelu** alle on merkitty muutamat projektin jäsenten pitämät suunnittelupalaverit sekä henkilökohtaisia suunnittelutehtäviä, jotka muodostavat tunneista suurimman osan. Suunnittelupalaverieita pidettiin projektissa erittäin vähän, sillä ajankohtaisten ratkaisemattomien asioiden selvittäminen tehtiin projektihuoneessa yleensä heti yhdessä.

Erillisestä **testauksesta** vastasi Tero, joka laati järjestelmätestaukselle suunnitelman ja suurimmaksi osaksi suoritti sen. Sovelluksen ajateltiin projektin alussa olevan prototyyppi, joten järjestelmätestausta ei edes ajateltu projektin alussa tehtävän.

Ryhmä auttoi Teroa muiden tehtävien ohessa tarvittaessa.

Toteutuksesta vastasi projektissa pääosin Juho, Marko ja Tero. Heikki osallistui projektipäällikkönä kuitenkin projektin alussa *tukitehtävät*-kohdan alle merkittyyn palvelinympäristön asentamiseen sekä projektin lopussa tehtyyn ldap-autentikointiin. Kehitystehtävien jakautuminen on nähtävissä suoraan tunneista, minkä lisäksi niitä on analysoitu myös luvussa 10.

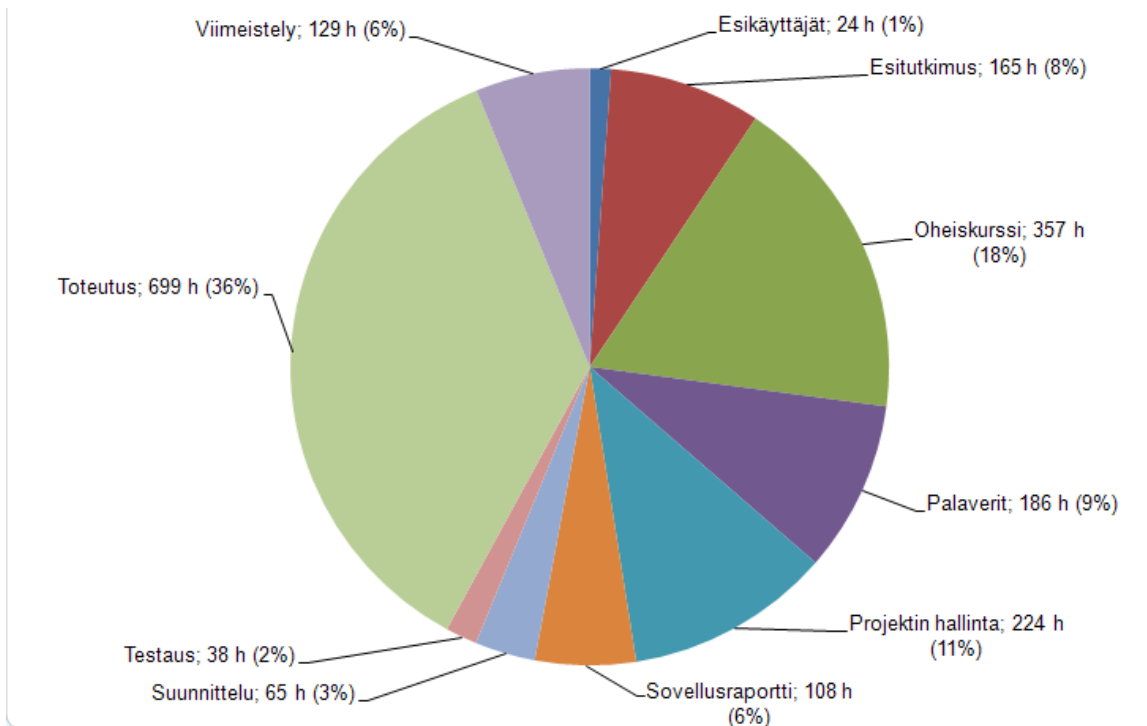
Viimeistelystä vastasi projektin lopussa pääosin Marko ja Tero Juhon ja Heikin laatiessa raportteja. Viimeistelyyn kirjatusta tunneista suurin osa koostuu toimivan sovelluksen lähdekoodin siistimisestä etenkin teknisen ohjaajan palautteen perusteella. Heikin muutamaiset tunnit lähdekoodin siistimisessä tulevat LDAP-toteutuksen siistimisestä.

10.2 Ryhmän ajankäyttö tehtäväkokonaisuuksittain

Verso-projektin ajankäytön jakautuminen tehtäväkokonaisuuksittain on esitetty kuvassa 10.2. Verso-projektilta tilattiin prototyyppi, minkä johdosta ohjelmiston kehittämiseen valittiin projektissa toteutuskeskeinen prosessimalli. Testaukseen (2 %) ja suunnitteluun (3 %) merkittyjen tuntien pieni määrä johtuu kirjauskäytänteistä. Vaatimusten toteuttamisesta valitun alustan päälle laadittiin suunnitelmia pääasiassa lähdekoodia lukien ja keskustellessa samalla kokeilla. Myös ohjelmointi ja testaaminen kulkivat kehitystyössä käsikkäin.

Oheiskurssiin (18 %) on merkitty Verso-projektin ohessa järjestetyille oheiskursseille kuuluvat luennot sekä ohjaajan edellyttämien korjausten tekeminen projektissa laadittuihin dokumentteihin. Kuitenkin, projektisuunnitelmaa kirjoitettaessa tunteja ei vielä kirjattu oheiskurssille, joten projektisuunnitelman korjausten tekemiseen kulunut aika on kirjattu projektin hallintaan.

Kukin jäsenistä toimi projektin aikana projektipalavereissa kahdesti sekä sihteerinä että puheenjohtajana. Palaverihin on kirjattu palaverien esityskalvojen tekeminen, palaveriin osallistuminen sekä sihteerin palaverista laatiman pöytäkirjan ensimmäisen version laatimiseen kulunut aika. Pöytäkirjoihin sen jälkeen ohjaajien palautteen perusteella toteutetut muokkaukset on sen sijaan kirjattu tehtäväkokonaisuuden *Oheiskurssi* alle.

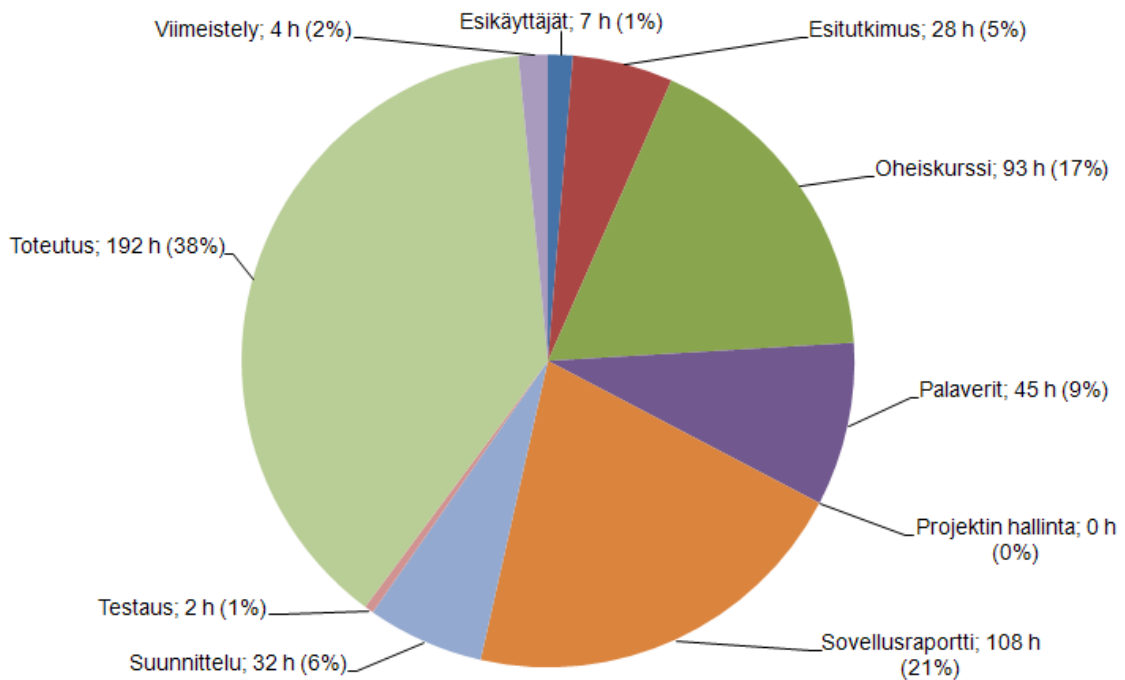


Kuva 10.2: Ryhmän ajankäyttö tehtäväkokonaisuuksittain.

10.3 Juho Niemisen ajankäyttö tehtäväkokonaisuuksittain

Juho toimi Verso-projektin varaprojektipäällikkönä. Juhon vastuulla oli sovellusraportti [16] sekä projektin alussa aloitettu vaatimusmäärittelydokumentti, jonka sisältö kuitenkin myöhemmin siirrettiin Trac-järjestelmään. Vaatimusmäärittelyn valmistelu näkyy Juholla kuvassa 10.3 muita suurempana suunnitteluun kuluneena aikana. Juho laati käytettävyydestauksien muistiot [12] ja [14].

Verso-projekti oli toteutuspainotteinen, mikä näkyy myös Juholla suurena toteutukseen käytettynä tuntimääränä (201 tuntia). Vaikka YouSource-sovelluksen ominaisuuksia kehitettiin pääosin ilman niiden henkilöitymistä jollekin projektin kolmesta kehittäjästä, Juholle erityinen osa toteutusta ovat YouSourcen ulkoasuun ja käytettävyyteen liittyvät parannukset. Juholla oli jo ennen projektia hyvät tiedot WWW-sovelluksen käyttöliittymän kehittämisestä, mistä YouSource lopputuloksena hyötyi.



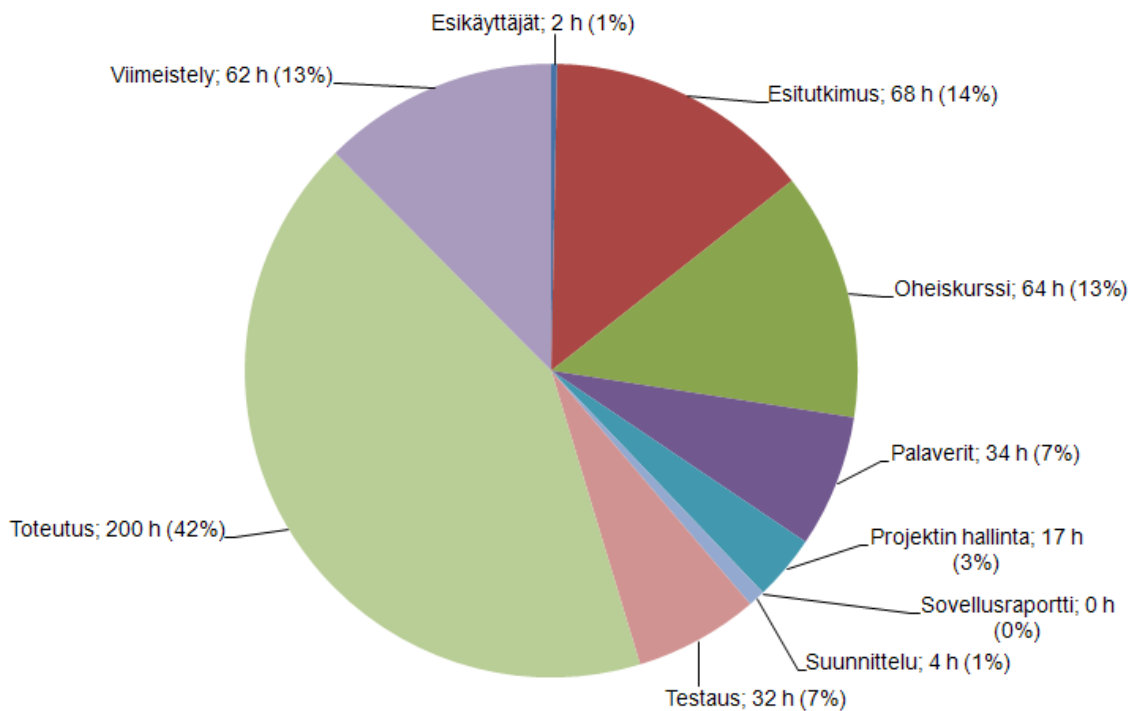
Kuva 10.3: Juhon ajankäyttö tehtäväkokonaisuuksittain.

10.4 Tero Hännisen ajankäyttö tehtäväkokonaisuuksittain

Teron ajasta kolme neljännestä kului esitutkimukseen, sovellusta kehittämiseen ja viimeistelyyn. Verso-projektin toteutuspainotteisuus näkyy myös Terolla suurena toteutukseen käytettynä aikana (200 tuntia).

Teron erityisesti Juhoa ja Heikkiä suurempi osuus esitutkimuksesta liittyy Git-versiohallintajärjestelmän rakenteeseen perehtymiseen. Teron vaikutus näkyykin eniten YouSourcen Git-versiohallintaohjelmiston toiminnallisuuksissa. Tero ei ollut aiemmin käyttänyt versiohallintaohjelmistoja, mutta projektin loppuessa Terolla oli Gitin rakenteesta selkeästi paras tuntemus.

Tero laati järjestelmätestaukseen liittyvät dokumentit [23], [24] ja [25], käytettävyysspäivän muistion [22], ja alustavertailun [26], sekä kokosi projektin kansion.



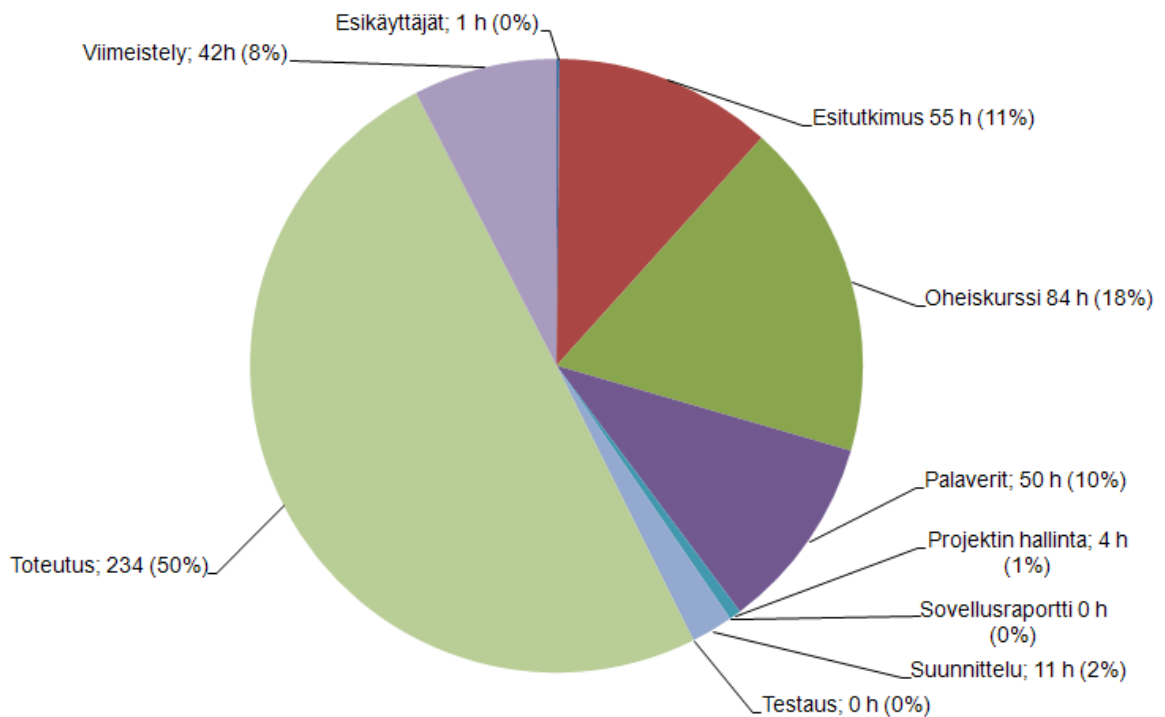
Kuva 10.4: Teron ajankäyttö tehtäväkokonaisuuksittain.

10.5 Marko Peltolan ajankäyttö tehtäväkokonaisuuksittain

Markon ajasta kului Teron tavoin melkein kolme neljännestä esitutkimukseen, sovelluksen kehittämiseen ja viimeistelyyn. Markokin käytti huomattavan suuren tuntimäärän sovelluksen kehittämiseen (234 tuntia).

Markon esitutkimukseen kulunut aika sisältää erityisesti projektissa käytetyn Ruby on Rails -sovelluskehityksen toimintaan perehtymistä. Markon vastuulla olivat YouSource-sovellukseen toteutettu näkyvyystasojen hallinta sekä projektin käytössä olleiden palvelinten `versotest.it.jyu.fi` ja `versoinstall.it.jyu.fi` konfigurointi esikäytön ajan ja asennusohjetta varten.

Marko laati YouSource-sovellukselle asennusohjeen [19] ja 2. koodinkatselmoinnin muistion [17], sekä koosti Verso-projektin dokumentit sisältävän WWW-sivuston.



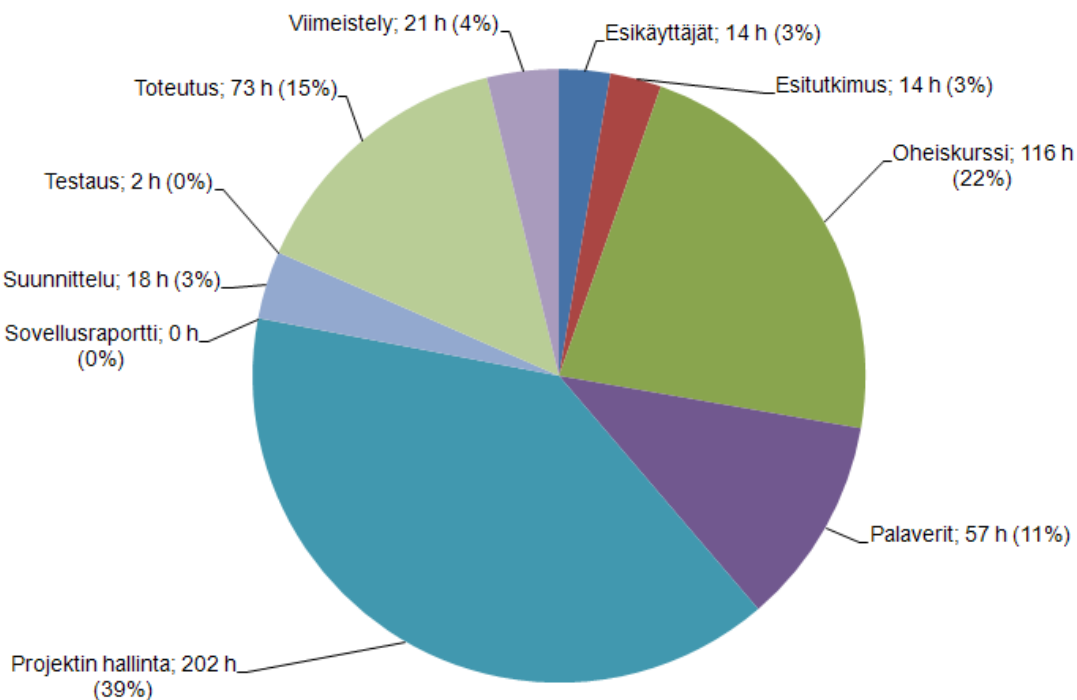
Kuva 10.5: Markon ajankäyttö tehtäväkokonaisuuksittain.

10.6 Heikki Salon ajankäyttö tehtäväkokonaisuuksittain

Heikin Verso-projektin projektipäällikkyyks näkyy korostuneena ajankäyttönä projektinhallinnassa (202 tuntia), josta valtaosa kului projektisuunnitelman ja -raportin laatimiseen.

Vaikka viivästyneen projektisuunnitelman laatiminen lopulta keskeytettiin, sen laatimisen osuus (91 tuntia) on huomattava, sillä se on noin seitsemäsosa Heikin projektiin käyttämästä ajasta projektipäällikkönä. Projektipäällikön muut velvollisuudet odottivat usein palvelinten konfiguroinnin tai LDAP-autentikoinnin toteuttamisen aikana.

Esikäyttäjien kanssa toimimiseen kului projektissa odotettua vähemmän aikaa, josta suurin osa kului projektipäällikön viestintään ja suunnitelman [9] laatimiseen. Heikki laati palaverien tilakatsauksen lisäksi projektiraportin [8], projektisuunnitelman [7], 1. koodikatselmoinnin muistion [6], tarvekuvauksen Korppi-ryhmien käyttämisestä [10] sekä KorppiLDAP:in käyttöä kuvaavan dokumentin [11].



Kuva 10.6: Heikin ajankäyttö tehtäväkokonaisuuksittain.

11 Prosessimalli ja projektin aikataulu

Projekti oli määrä toteuttaa kevätlukukaudella 2010 siten, että sovellus olisi ollut uusien ominaisuuksien toteuttamisen osalta valmis 23.4.2010 mennessä ja kokonaisuudessaan toukokuun puoleen väliin mennessä. Uusien ominaisuuksien kohdalla aikataulu piti, mutta projektin dokumentointi viivästyi noin puolella vuodella suunnitellusta toukokuun lopusta. Aikataulua tarkastellaan tarkemmin luvussa 11.2.

Suunniteltuun verrattuna merkittävimmät aikataulun erot liittyivät projektin raportoinnin viivästymiseen sekä projektipäällikön suunniteltua suurempaan teknisten tehtävien tekemiseen. Edellämainitut kaksi asiaa ovat myös kytköksissä keskenään. Projektisuunnitelman [7] viivästyminen ja jäädyttäminen keskeneräisenä hankaloitivat projektin läpivientä sekä tekivät sen toteutumisen analysoimisesta vaikeaa tai paikoitellen mahdotonta.

11.1 Prosessimalli

Projekti oli määrä viedä läpi käyttäen ketterää ohjelmistokehitystä. Kesken jääneessä projektisuunnitelmassa [7] määriteltiin projektissa käytettävän projektille räätälöityä ketterää prosessimallia. Suunnitelmassa määriteltiin käytettävän kahden viikon mittaisia toteutusvaiheita, joiden kehitystehtävät sovittiin kunkin vaiheen alussa pidettävässä projektipalaverissa projektiryhmän tekemän ehdotuksen pohjalta.

Projektiryhmän muodostama **ehdotus toteutusvaiheen tehtävistä** muodostui palaverikeskustelujen, kiinnittämättömien tehtävätikettien ja muun yhteydenpidon perusteella. Projektiryhmä valitsi vaiheeseen kiinnitettävät tehtävät käytettävissä olevat resurssit huomioiden. Tehtävien kokojen arvioinnissa onnistuttiin pääosin hyvin, vaikka ensimmäisistä toteutusvaiheista jäi noin päivän tai kahden edestä valmistumattomia ominaisuuksia. Valmistumattomat ominaisuudet hyväksyttiin palaverissa siirrettäväksi seuraavan vaiheeseen sen alussa.

Projekti vietiin läpi käyttäen **ketterää ohjelmistokehitystä**. Vaatimusmäärittelyn ominaisuus eli Trac-tikettityyppi Feature hyväksyttiin valmistuttuaan tilaajalla. Hyväksyttäminen tehtiin tavallisesti siten, että projektiryhmä järjesti ominaisuuden testipalvelimelle ja ilmoitti siitä projektin sähköpostilistalla. Pari kertaa tilaaja ei hyväksynyt kehitettyä ominaisuutta sellaisenaan, jolloin projektiryhmä korjasi ominaisuudesta löytyneet puutteet seuraavan toteutusvaiheen aikana.

Kehitystyössä käytettiin **neljää kahden viikon mittaista toteutusvaihetta**, joiden lisäksi viimeistelylle oli varattu viides kolmen viikon mittainen viimeistelyvaihe. Viidennessä vaiheessa ei enää kehitetty uusia ominaisuuksia, ellei niiden kehittämistä oltu jo aloitettu aiemmissa vaiheissa.

Projektille ominainen piirre oli **tehtävien määrästä sopiminen vaiheiden aikataulun perusteella**. Käytännössä Verso-projektin sovelluksen kehittämiseen oli alusta asti varattu neljä kehitysvaihetta, joihin sisältyvät tehtävät valikoituivat edellisen vaiheen aikana. Kyseisen käytänteen esittäminen ei olisi ollut lainkaan mielekäästä, jos projektiryhmän motivaatio ei jo alusta asti olisi näyttänyt korkealta. Tasainen ja loppuun asti kestänyt kehitysrytmi ei kuitenkaan ollut itsestäänselvyys, ja projektin lyhyen keston vuoksi selkeämpi valinta olisi ollut tarkastella sovelluksen ominaisuuksia alusta asti inkrementteittäin. Verso-projektiin sovitut vaiheet muistuttivat kuitenkin enemmän aikataulutettuja kehityspyrahdyksiä (engl. *sprint*).

Prosessimallin toteuma vastasi hyvin suunniteltua, joskin projektisuunnitelman [7] prosessimallikuvaus jäi suppeaksi. Projektisuunnitelmassa kuvatut projektin Trac-tikettikäytännöt toteutuivat lähes sellaisenaan.

Projektissa toteutetulle **sovellukselle ei laadittu erillistä vaatimusmäärittelyä**, vaan vaatimuksia ylläpidettiin tiketteinä Trac-järjestelmässä. Vaatimusmäärittelyn osalta projektissa nojattiin liiankin paljon tiketteihin etenkin, kun yksittäisiä tikettejä ylläpidettäessä kokonaiskuva jää määrittelemättä. Lisäksi projektin kesto oli vain hädin tuskin tarpeeksi pitkä hyödyn saamiseksi ketteristä päätöksistä eli projektin aikana muodostuneiden tarpeiden pohjalta tehdyistä valinnoista toteutusvaiheiden tehtäviä kiinnitettäessä. Trac oli työkaluna erityisen huono valinta kokonaisuuksien hahmottamiseen tikettien pohjalta, mikä hankaloitti valmiin sovelluksen ominaisuuksien hahmottamista.

Koska projektin alussa **tilaajalla ei ollut tarkkoja vaatimuksia** sovelluksen ominaisuuksien tai alustavalintojen osalta, olisi jonkinlainen kokoavampi vaatimusmäärittely auttanut suunnittelua. Tässä olisi päässyt pitkälle jonkinlaista tuotteen työllistää (engl. *product backlog*) tukevalla tikettijärjestelmällä, joka mahdollistaisi tehtävien suunnittelun korkeammalla tasolla. Tracin eduksi voi kuitenkin todeta sen toimineen yksittäisiä tehtäviä ajatellen hyvänä keskustelu-ympäristönä projektissa.

Tikettien kestoja arvioitiin ensimmäisen kolmen kehitysvaiheen ajan vain kokonaisuutena valittaessa kehitystehtäviä seuraaviksi kahdeksi viikoksi. Arviot pitivät paikkansa kohtuullisesti, sillä yleensä vaiheelle valittuja tehtäviä jäi tekemättä yhden tai kahden kehityspäivän edestä.

Neljäs kehitysvaihe oli viimeinen, jolle otettiin uusia kehitystehtäviä. Projektiryhmä arvioi sen alussa jäljelläolevien potentiaalisesti tilaajan tärkeiksi katsomien tikettien kestot tunteina Traciin. Tämä osoittautui hyödylliseksi, kun neljännen vaiheen aloitavassa projektipalaverissa valittiin, mitkä ominaisuudet kehitetään ja mitkä rajataan jatkokehitykseen.

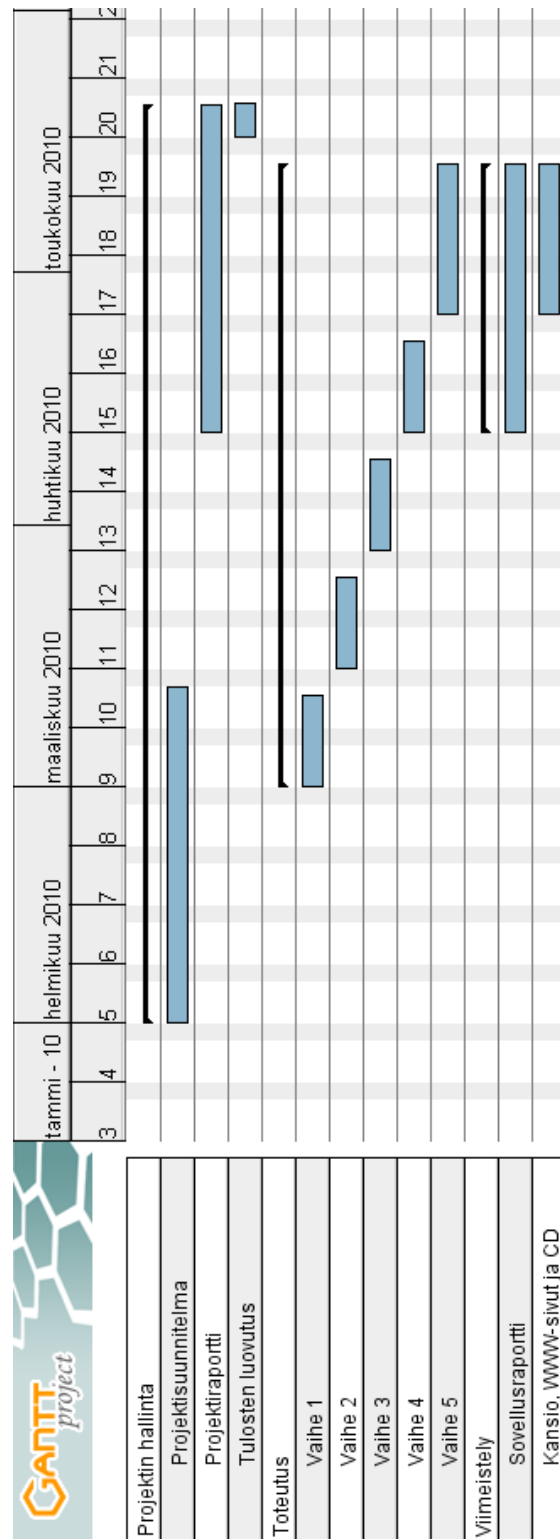
Projektiryhmän neljännelle vaiheelle arvoimat tikettien kestot pitivät keskimäärin paikkansa hyvin. Kun KorppiLDAP:in toteuttaminen ja näkävyyksien muokkaaminen vaativat arvioitua enemmän tunteja, muut valitut tehtävät valmistuvat nopeammin.

11.2 Aikataulu

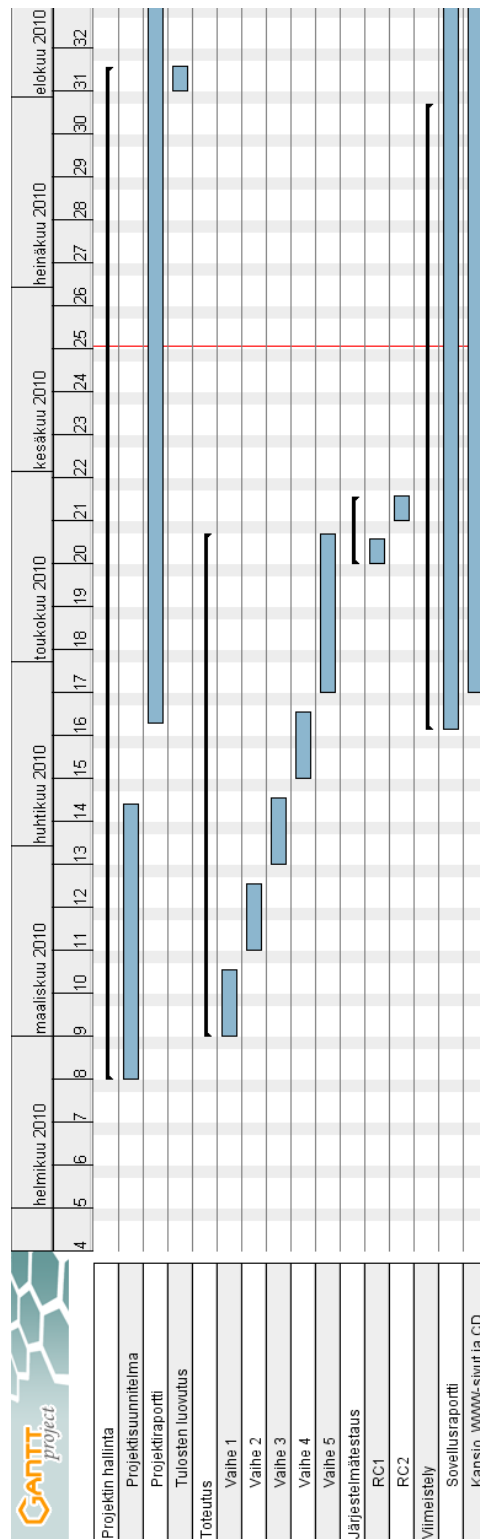
Projektin oli määrä olla kokonaisuudessaan valmis vuoden 2010 toukokuun puoleen väliin mennessä, mutta se myöhästyi siitä noin puolella vuodella. Sovelluksen osalta kehitystyö viivästyi venyneestä viimeistelystä johtuen noin kahdella viikolla.

Kuvan 11.1 suunnitellun ja kuvan 11.2 toteutuneen aikataulun välillä **merkittävin ero liittyy viimeistelyyn ja raportointiin**, erityisesti raportointiin. Projektisuunnitelma viivästyi suunnitellusta, ja sen laatiminen loppujen lopuksi keskeytettiin. Sovellusraportti hyväksyttiin 20. syyskuuta. Projektiraportin laatiminen jatkui kuvan ulkopuolella joulukuulle asti. Sovellus- ja projektiraporteille olisi pitänyt varata huomattavasti suunniteltua enemmän aikaa. Järjestelmätestaus oli projektin lopussa otettu tehtävä, johon kului myös aikaa. Tämä ei viivästyttänyt suoraan projektin pahiten myöhästyneitä osia, mutta viivästytti tulosten luovuttamiseen liittyviä tehtäviä. Jälkikäteen voi nähdä, että pysyäkseen aikataulussa myös raporttien ja viimeistelyn osalta Verso-projektista olisi pitänyt karsia kehitystehtäviä vähintään yhden toteutusvaiheen verran.

Kuvissa 11.1 ja 11.2 ei näy kehitystehtävien vaiheittaista myöhästymistä, sillä kunkin vaiheen myöhästyneet ominaisuudet hyväksyttiin osaksi seuraavaa vaihetta. Vaiheet toteutuivat kuitenkin pääosin niiden alussa suunniteltuina, sillä kehitystehtäviä jäi tekemättä korkeintaan noin päivän tai kahden edestä kunkin kehitysvaiheen aikana.



Kuva 11.1: Projektin läpiviennin suunniteltu aikataulu.



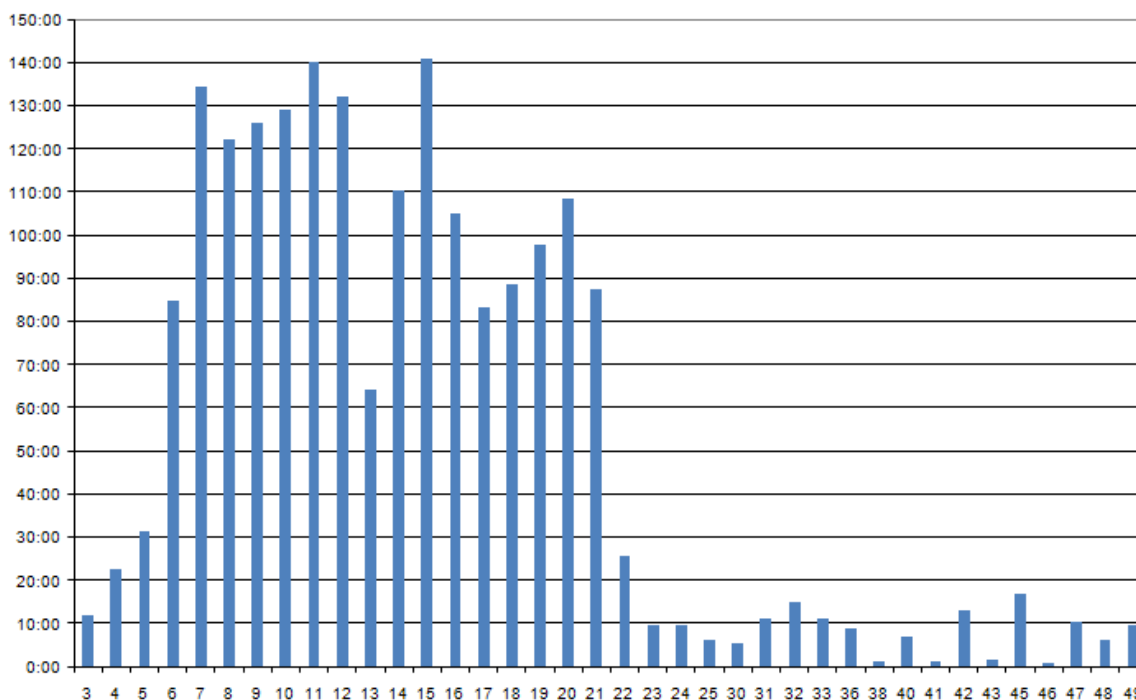
Kuva 11.2: Projektin läpiviennin toteutunut aikataulu.

11.3 Ryhmän ajankäyttö viikoittain

Ryhmä käytti projektiin paljon työtunteja, ja kukin neljästä jäsenestä teki kehitysvaiheiden aikana enimmillään yli 30 tunnin keskimääräistä viikkoa.

Viikottaisen ajankäytön kuvassa 11.3 näkyy selvästi neljä trendiä:

- projektin alku ennen tilaajaa ja toimeksiantoa (viikot 3-5)
- kehityksen alkupuoli (viikot 5-15)
- kehityksen loppupuoli (viikot 16-21) ja
- viimeistely (viikosta 21 ja eteenpäin).



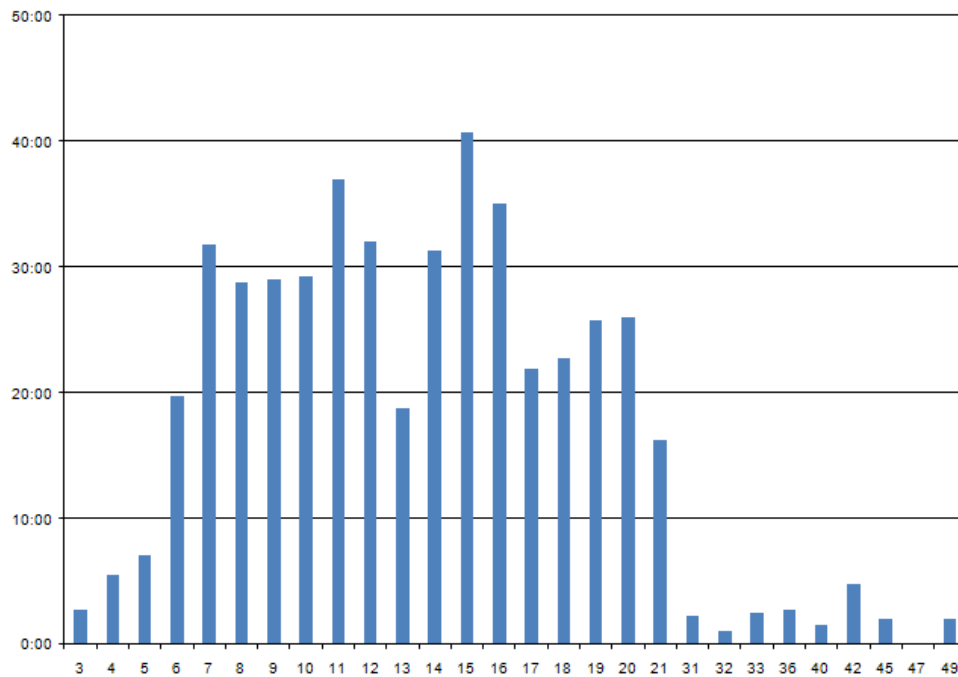
Kuva 11.3: Ryhmän ajankäyttö viikoittain.

Projektin kolmen ensimmäisen viikon tunnit tulevat yksinomaan oheiskursseihin liittyvistä luennoista. Tilaajan ja toimeksiannon selvittyä projekti käynnistyi nopeasti viikosta 5 eteenpäin, ja aihealueeseen perehdyttiin paljon tunteja käyttäen, johon tarvetta loi ryhmälle verrattain vieras aihealue. Kevään loppupuolella (viikot 16-21) tuntimääriä tietoisesti vähennettiin, koska jäsenet olivat käyttäneet projektiin tunteja omaa mukavuustasoaan enemmän. Lisäksi sovelluksen nähtiin lisäksi voivan valmistua pienemmälläkin tuntimäärillä. Sovelluksen kehitystä olisi pitänyt kuitenkin karsia dokumentoinnille tilaa antaen etenkin, kun ryhmän jäsenten kesätyöt alkoivat viikolla 22.

Viikoilla 13 ja 14 näkyy Juhon sairastumisen ja pääsiäispäivien aiheuttama vaje tunneissa. Myös ryhmän kesätöiden alkaminen näkyy selvästi romahduksena tunneissa viikolla 22. Kuvassa 11.3 ei edes näytetä niitä viikkoja kesältä ja syksyltä, jolloin työtunteja ei kertynyt.

11.4 Tero Hännisen ajankäyttö viikoittain

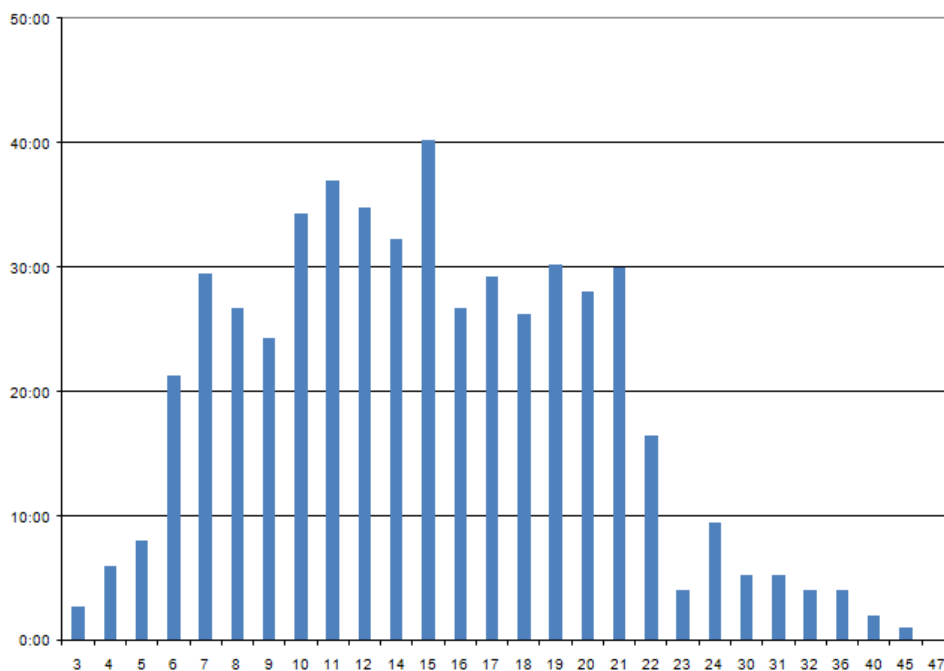
Ryhmän jäsenten viikoittaiset työtunnit olivat pääpiirteissään yhteneviä. Tero jäi jatkokokehittämään projektissa kehitettyä ohjelmistoa viikolla 21, mikä näkyy tuntien määrän vähenemisenä. Tero vastasi projektin kansion ja sähköpostiarkistojen koostamisesta, mikä näkyy kesäkauden aikana tehtyinä tunteina.



Kuva 11.4: Teron ajankäyttö viikoittain.

11.5 Juho Niemisen ajankäyttö viikoittain

Juhon ajankäyttö noudattelee myös ryhmän ajankäyttöä. Kuvasta 11.5 puuttuu Juhon sairastumisen takia kokonaan pääsiäisviikko 13. Juhon kesätyö alkoi viikolla 21, mikä näkyy graafissa tuntimäärän vähenemisenä. Juhon vastuisiin projektin loppupuolella kuului sovellusraportin laatiminen, mikä näkyy Teroa ja Markoa suurempana ajankäyttönä loppukesällä ja syksyllä.

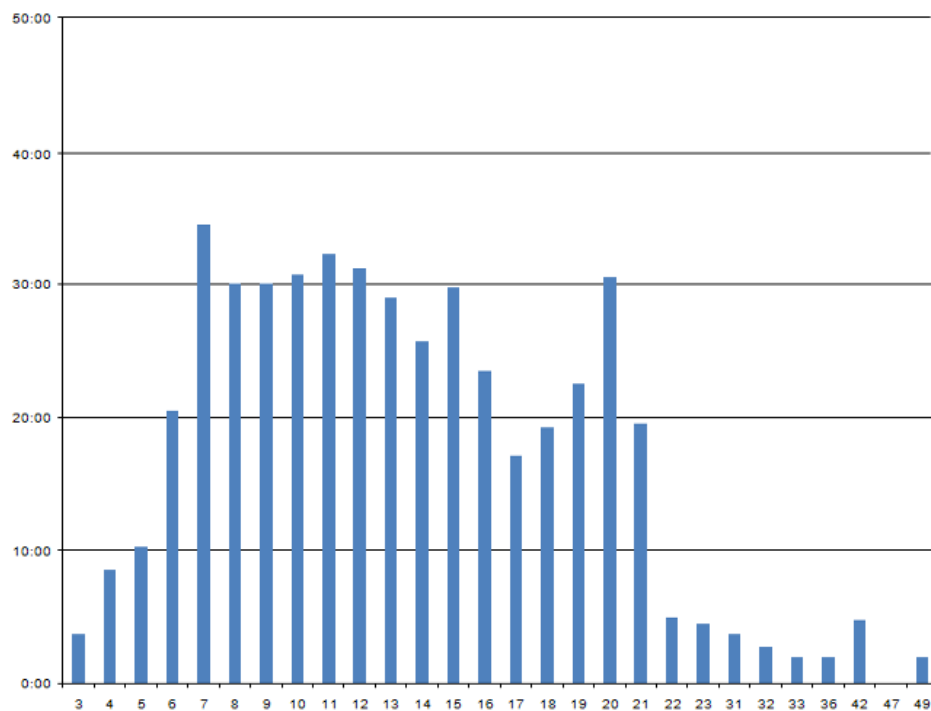


Kuva 11.5: Juhon ajankäyttö viikoittain.

11.6 Marko Peltolan ajankäyttö viikoittain

Myös Markon ajankäyttö noudattelee ryhmän yleistä ajankäyttöä. Marko aloitti Teron kanssa projektissa toteutetun sovelluksen jatkokehityksen viikolla 21, mikä näkyy tuntimäärien vähenemisenä. Markon vastuisiin kuului projektin loppupuolella Verso-projektin verkkosivuston kokoaminen palvelimelle

<http://sovellusprojektit.it.jyu.fi/> ja sovelluksen asennusohjeen viimeistely.

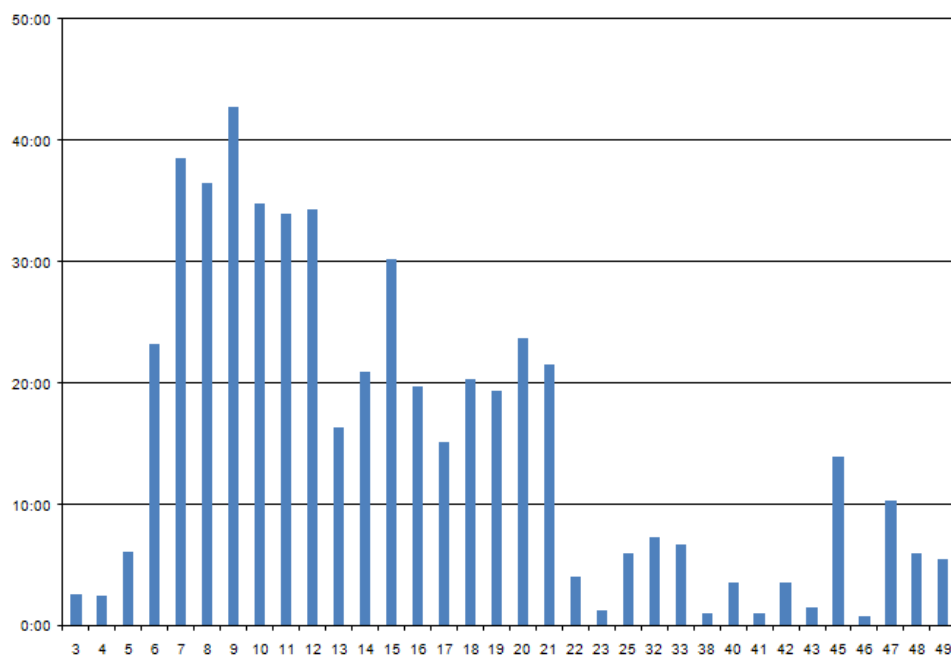


Kuva 11.6: Markon ajankäyttö viikoittain.

11.7 Heikki Salon ajankäyttö viikoittain

Heikki oli Verso-projektin projektipäällikkö, mikä näkyy muuta ryhmää lievästi suurempana ajankäyttönä projektin alkupuolella. Projektin alussa Heikki suoritti projektipäällikön tehtävien ohessa myös järjestelmien asentamista.

Myös Heikin kesätyö alkoi viikolla 21, mikä näkyy viikoittaisten tuntimäärien vähenemisenä. Heikillä oli projektin loppuunsaattamisessa organisointitehtävien ohella laadittavana projektiraportti, mikä näkyy käytettyinä tunteina kesällä ja syksyllä.



Kuva 11.7: Heikin ajankäyttö viikoittain.

12 Riskit ja niiden seuranta

Luvussa kuvataan projektin ennakoitujen ja ennakoimattomien riskien toteutumista sekä vaikutuksia tuloksiin tai projektiin. Keskeneräinen projektisuunnitelma [7] kuvasi suppeasti riskejä ja niiden ennakointia. Ennustetut riskit toteutuivat joko keskinkertaisina tai pieninä.

Keskinkertaisina toteutuneen riskin *muutostarve kehityskoneissa* vaikutus projektin läpivientiin oli lopulta vähäinen. Riski *jäsenten motivaation puute* sen sijaan toteutui vaikuttaen projektin läpivientiin ja viivästyttäen sen valmistumista. Vaikka kevätkauden aikana projektiryhmän motivaatio ja omistautuminen olivat erinomaisia, projektin dokumentaatio valmistui noin puoli vuotta myöhässä suunnitellusta, koska kesän ja syksyn aikana projektin dokumentaation laatimiseen ei käytetty tunteja juurikaan.

12.1 Riskien todennäköisyydet ja haitat

Riskien todennäköisyydet ja niistä seuraavat haittavaikutukset on esitetty taulukossa 12.1. Todennäköisyyttä ja haittavaikutusta arvioidaan asteikolla pieni, keskinkertainen ja suuri.

Riski	Todennäköisyys	Arvioitu haitta	Toteutunut haitta
Muutostarve kehityskoneissa	suuri	keskinkertainen	keskinkertainen
Jäsenten motivaation puute	pieni	keskinkertainen	keskinkertainen
Valitun alustan ongelmat	keskinkertainen	suuri	pieni
Jäsenten poissaolot	keskinkertainen	pieni	pieni
Testipalvelimen ongelmat	keskinkertainen	keskinkertainen	pieni
Esikäyttäjien aktiivisuus	keskinkertainen	(ei arviota)	pieni

Taulukko 12.1: Riskit, todennäköisyydet ja haitat.

12.2 Muutostarve kehityskoneissa

Projektiryhmän käytössä oleviin kehityskoneisiin oli asennettu ryhmän tarvitsemista kehitystyökaluista ja sovelluskirjastoista tärkeimmät projektin kahden ensimmäisen viikon kuluessa. Projektin edetessä työasemiin tarvittiin kuitenkin muutoksia esimerkiksi *git pusheja* tai LDAP-autentikointia sovellukseen toteutettaessa. Uusien työkaluvaatimuksien kanssa projektiryhmä oli riippuvainen ATK-lähituen ylläpidosta, koska projektiryhmällä ei ollut kehityskoneisiin ylläpito-oikeuksia. Riskiksi arvioitiin, että projektiryhmä joutuu odottamaan uusia työkaluja.

Riskiä yritettiin minimoida pyytämällä tiedossa olevia työkaluja parhaimmillaan noin viikko tai kaksi etukäteen. Riski toteutui pahiten sellaisissa tilanteissa, joissa tarvittavaa muutosta kehityskoneisiin ei saatu lainkaan projektin aikana. Esimerkiksi juuri *git pushia* tai Python-kielen LDAP-laajennusta ei saatu projektin aikana toimimaan kehityskoneilla. Riskin vaikutus jäi näissäkin tilanteissa korkeintaan keskinkertaiseksi, sillä projektiryhmä kehitti tavat kokeilla ominaisuuksia virtuaalipalvelimen `versotest.it.jyu.fi` kautta.

Osa projektin aikana kehityskoneiden ylläpitoon liittyvistä vaikeuksista johtui siitä, että kehityskoneiden ja testipalvelimen ohjelmapakettivarastoissa oli samoista ohjelmista eri versioita, jotka toimivat hieman eri tavalla. Eri versiot vaikeuttivat myös sovelluksen asennusohjeen kirjoittamista.

12.3 Jäsenten motivaation puute

Riskiksi arvioitiin, ettei projektiryhmän yleinen motivaatio ole projektin aikana tarvittavalla tasolla projektille asetettujen tavoitteiden saavuttamiseksi. Riskin toteutumisen todennäköisyys arvioitiin pieneksi. Projektin alussa projektiryhmä koostui toisilleen täysin vieraista opiskelijoista, joita kohtasi projektin alussa jyrkkä oppimiskäyrä ryhmäkemian vasta muodostuessa.

Riski ei kevääseen sijoittuneen projektin varsinaisen elinkaaren aikana toteutunut lainkaan. Projektipääällikkönä tässä kohtaa on todettava, että suurempaa motivaatiota sovelluksen kehittämistä kohtaan on vaikea kuvitellakaan.

Projektiin kuuluvan dokumentoinnin kanssa on kuitenkin nähtävissä vähintään keskinkertaisia motivaatio-ongelmia, sillä projektin dokumentaatio on aiotusta vuoden

2010 toukokuun puolivälistä myöhässä noin puoli vuotta. Suuri osa "motivaatio-ongelmista" selittynee projektin jäsenten kesätöillä sekä poikkeuksellisen hienolla kesällä, mutta puolen vuoden myöhästyminen on silti laskettava keskinkertaiseksi motivaatio-ongelman toteutumiseksi.

12.4 Valitun alustan ongelmat

Sovelluksen alustaksi valikoitui Gitorious, koska se nähtiin lupaavimmaksi tavoitteiden saavuttamisen kannalta projektiryhmän suorittamassa alustavertailussa [26]. Projektin kuluessa oli silti mahdollista, että Gitoriousin käyttö alustana estyy tai hankaloituu jonkin ennustamattoman syyn perusteella. Se olisi voinut olla esimerkiksi Gitorious-sovelluksen jonkin sellaisen tietoturvaongelman ilmeneminen, joka estäisi sen käytön salaisten tietojen tai yliopiston käyttäjätunnuksien kanssa.

Riski ei liittynyt erityisesti Gitorious-sovellukseen tai sen pohjana toimivaan Ruby on Rails -sovelluskehikseen. Riski olisi toteutuessaan olennaisesti hankaloittanut sovellukselle asetettujen tavoitteiden saavuttamista. Riskin vaikutus arvioitiin pieneksi, ja sen vaikutus projektiin jäi myös pieneksi. Ryhmä korjasi Gitoriousesta yhden YouSourcea kehittäessä esiintulleen tietoturvapuutteen, joka korjautui samalla myös Gitorious-kantasovellukseen.

Riskiä ei varsinaisesti voitu kontrolloida, mutta projektiryhmä seurasi koko projektin ajan kaikkia Gitoriousin tiedotuskanavia ja olisi näin kuullut ongelmista nopeasti.

12.5 Jäsenten poissaolot

Riskiksi arvioitiin, että jokin ryhmän jäsen on pois ilman tarvittavia järjestelyjä, sekä muu projektiryhmä ei tietäisi poissaolevan toimien tilanteesta tai niiden hoitamisesta. Riskin toteutumisen todennäköisyys arvioitiin pieneksi. Hankalan poissaolon olisi voinut muodostaa sairastuminen, mutta myös lomamatka. Riskin vaikutus arvioitiin keskinkertaiseksi.

Riskin toteutumisen todennäköisyys arvioitiin pieneksi, sillä kellään ryhmästä ei ollut projektin alussa tiedossa viikonloppua pidempiä lomamatkoja keväälle, eikä sairastumisista johtuvia poissaoloja ollut aiheutta odottaa ennalta. Riskiä hallittiin pro-

jektin alusta loppuun kestäneellä hyvällä kommunikoinnilla jäsenten kesken. Lisäksi uusia toiminnallisuuksia toteuttaessaan ryhmän jäsenet toteuttivat ohjelmointia monilta osin pariohjelmointina, jota kautta tietämys uusista tekniikoista levisi heti tehdessä.

Riski ei toteutunut lähes lainkaan, vaikka Juho oli sairaana pari päivää ja Heikki päivän.

12.6 Testipalvelimen ongelmat

Riskiksi arvioitiin, että esikäytössä oleva sovellus olisi alkanut toimia tuntemattomasta syystä väärin. Projektissa toteutettava sovellus sisältää paljon erilaisia komponentteja, joiden yhteistoiminnasta sovellus on riippuvainen. Jos sovelluksen jokin osa olisi vikaantunut tai alkaisi toimia väärin, tilanteen selvittäminen olisi voinut olla haastavaa ja aikaavievää, mikä olisi viivästyttänyt kehitystyötä.

Riskin vaikutukset arvioitiin keskinertaisiksi sen projektia viivästyttävän luonteen vuoksi. Riskiä hallittiin projektin aikana järjestämällä testipalvelimen lokitiedostojen kirjoitus kuntoon, jotta mahdollisissa vikatilanteissa mahdollisimman paljon tietoa tapahtuneesta olisi käytettävissä myöhemmin. Luvussa 7.4 kuvataan lisäksi automaattisten virhetietoa sisältävien poikkeustiedotteiden käyttöä.

Riski toteutui vain pienenä, koska ongelmatilanteiden ratkaisussa ei yleensä mennyt pitkään. Projektisuunnitelmassa [7] kuvattiin riskin hallitsemiseksi harjoitteet erilaisten vikatilanteiden etukäteen harjoittamiseksi. Suunnitellusta kuitenkin poikettiin, sillä vastaantulleet ja mahdollisesti yleistyvät vikatilanteet korjattiin yleensä koko projektiryhmän voimin ja tietämys korjaustoimenpiteistä levisi saman tien. Tarvetta erilliselle harjoittamiselle ei siten lopulta ollut.

12.7 Esikäyttäjien aktiivisuus

Riskiksi arvioitiin, että projektissa toteutetun sovelluksen julkaiseminen esikäyttöön aiheuttaa projektiryhmälle niin suuren ylläpito- tai neuvontataakan, että se viivästyttää projektia. Riskin toteutumisen todennäköisyyttä tai suuruutta ei arvioitu etukäteen lainkaan, eikä se toteutunut, sillä esikäyttäjien aktiivisuus oli vähäinen, kuten luvussa 7.5 kuvataan.

13 Projektiryhmän jäsenten kokemuksia

Luvussa kuvataan projektin jäsenten kokemuksia Verso-projektista. Projekti onnistui laajuuteensa nähden hienosti. Projekti sisälsi ryhmän kannalta monia uusia asioita, kuten uudet tekniset työkalut, tilaajan kanssa toimiminen ja projektikäytänteet sekä ATK-tuen ja sekä sovelluspalveluiden kanssa toimiminen.

13.1 Mitä tekisimme toisin?

Suurin yksittäinen toive toisin tehtävästä liittyy versiohallintaan ja siinä projektin loppupuolella vastaantulleisiin ongelmiin, jotka kuvattiin parannusehdotuksineen luvussa 8.

Projektin vaatimusmäärittelyyn olisi projektin alussa voinut myös keskittyä enemmän. Ajatus pysyvien ja versiottomien (engl. *static*) tiedostojen jakamisesta muotoutui vasta 8. projektipalaverissa, vaikka se liittyi olennaisesti kehitettävään järjestelmään.

Kolmas muutostoive liittyy dokumentteihin. Esimerkiksi projektisuunnitelma olisi ehdottomasti ensin pitänyt luonnostella erillisiin pieniin ja helposti hallittaviin osiinsa, eikä yrittää tehdä siitä heti ladottua dokumenttia. Myös täysin keskeneräisiin dokumentteihin liittyvät sisältöä koskemattomat kieliasu- tai tyylihuomiot olisi pitänyt osata sivuuttaa täysin.

13.2 Tero Hännisen kokemuksia omin sanoin

Projektin alussa oli pientä jännitystä siitä, mitä tuleman pitäisi. Oppimiskäyrä oli siinä mielessä jyrkkä, että suurin osa työkaluista oli vieraita. Samasta syystä projekti oli erittäin opettavainen, sillä itse tulin oppineeksi todella paljon ainakin seuraavista aiheista: Git, Ruby, Ruby on Rails, WWW-kehitys ja Linux-terminaali. Mainitut työkalut ovat mielestäni fiksuja ja tehokkaita, joten olenkin todella tyytyväinen niiden oppimisesta tehokkaasti projektin myötä. On paljon mukavampaa opetella asioita, kun on todellinen tarve siihen ja mahdollisuus soveltaa kaikkea oppimaansa heti sen sijaan, että opettelisi vain oppiakseen.

Projekti onnistui mielestäni hienosti. Kaikki ryhmän jäsenet tekivät hommia hyvin,

yhteistyö sujui ja henki oli hyvä. Toisessa lähdekoodin katselmoinnissa tekninen ohjaaja kehui koodia varmaotteiseksi opiskelijoiden kirjoittamaksi. Lopputulos ylitti tilaajan tavoitteen saada jonkinlainen prototyyppi, kun tuloksena olikin toimiva ja käytettävä sovellus. Tähän toki auttoi se, että sovelluksen pohjaksi löytyi paljon toimintoja tarjoava Gitorious.

Projektin läpiviennin kannalta kriittistä oli avoin ja aktiivinen viestintä. Tässä ei mielestäni ollut ongelmia, mutta erityisesti projektipäällikkömme Heikki hoiti viestintää vahvasti sekä tilaajien ja ohjaajien että tietohallintokeskuksen palveluiden suuntaan. Toki viestintä toimi hyvin myös toiseen suuntaan. Myös projektiryhmän sisällä tieto kulki todella hyvin kaikkien työskennellessä samassa huoneessa ja perustamamme IRC-kanavan ansiosta.

13.3 Juho Niemisen kokemuksia omin sanoin

Ennen sovellusprojektin alkua olin kuullut, että kurssin on työläs, eikä muita kursseja ehdi oikeastaan suorittaa sen aikana. Olin myös kuullut, että jos ohjaajaksi osuu Jukka-Pekka Santanen, niin kurssin työläin osio tulee olemaan dokumenttien laatiminen. Nämä molemmat kuulopuheet osottautuivat oikeiksi oman sovellusprojekti kurssini kohdalla. Projektihuoneessa tuli vietettyä aikaa jokaisena mahdollisena työpäivänä, joten muiden kurssien suorittaminen jäi iltatöiksi. Ja kuten arvata saattaa, ei sellaisesta hyvää seuraa. Dokumenttien laatiminen oli kurssilla todellakin iso urakka, mutta ei aivan niin iso kuin ennalta pelkäsin.

Sovellusprojektin aikana oli hyvä meininki. Sovellusta tehtiin innolla, ja ominaisuuksia kehitettiin ihan omasta aloitteesta. Pieniä korjauksia tehtiin sinne tänne vain tekemisen ilosta. Valitettavasti joitain isojakin puutteita sovelluksen viimeiseen versioon jäi, joten työn kohdentamista olisi voinut harkita tarkemmin (eikä vain suorittaa niitä kivoja hommia).

Projektin lopussa pidettiin loppuesittely, joka oli hyvä ja opettavainen kokemus puheviestinnän kannalta. Projektista jäi hyvä mieli, kun huomasi, että muitakin kiinnostaa projektissa tehty työ. Loppujen lopuksi koko sovellusprojekti oli hyvin positiivinen kokemus, ja siitä oppi paljon. Git, Ruby on Rails, Trac, Linuxin komentokehote, palaveripöytäkirjat, sovellusraportti ja käytettyyystestaukset ovat kaikki uusia asioita, jotka tulivat tutuksi projektin aikana. Kokemus ei varmasti olisi ollut niin hyvä, jos projektin aihe ei olisi ollut niin mielenkiintoinen, asiakas olisi ollut

hankala tai projektiorganisaatio ei olisi tarjonnut niin hyviä tiloja ja palvelua projektiryhmälle.

13.4 Marko Peltolan kokemuksia omin sanoin

Ennen sovellusprojektia olin kadottanut kiinnostukseni ja motivaationi tietotekniikkaa ja opiskelua kohtaan. Tentteihin pännääminen ja turhilta vaikuttavien harjoitustöiden tekeminen pelkästään opintopisteiden takia ei ole innostanut minua. Olen kaivannut enemmän käytännönläheisempiä opintoja, ja sovellusprojektin aikana sain tuntumaa, mitä ohjelmistokehitys oikeasti voisi olla.

Ennen projektin alkua hieman jännitti, minkälaisen ryhmän ja sovelluksen kanssa muutaman kuukauden projekti kuluu. Ryhmä osoittautui mitä mainioimmaksi, ja projektin aihekin oli mielenkiintoinen. Olin toivonut sovellusprojektin aiheeksi WWW-sovellusta, jollaisen kehittäminen tehtäväksemme tulikin. Lähtökohdat projektiin olivat kaiken kaikkiaan hyvät.

Aluksi oli paljon uutta opeteltavaa versiohallinnasta, uudesta ohjelmointikielestä ja uudesta ohjelmistokehityksestä, mutta kaikki ryhmän jäsenet olivat koko projektin ajan motivoituneita tekemään töitä. Päivät olivat välillä pitkiä, ja ennen sovellusprojektia varoiteltu suuri työmäärä toteutui. Projekti oli kuitenkin niin mukava, että istuin mieluummin projektihuoneessa kuin luennoilla, ja muut kurssit saivat väistyä projektin tieltä odottamaan parempia aikoja.

Olisin toivonut, että sovellusprojektin aikana olisimme saaneet enemmän ohjausta itse sovelluskehitykseen. Ohjauksen painopiste oli dokumenttien kieliasun tarkastamisessa, vaikka sitä varten oli tarjolla oma oheiskurssinsakin. Tekniseltä ohjaajalta saimme hyviä neuvoja silloin, kun häneltä neuvoa tajusimme kysyä. Varsinkin projektin alkuvaiheessa olisimme tosin voineet käyttää hänen asiantuntemustaan enemmänkin hyödyksi.

Sovellusprojekti oli mielestäni hyvin onnistunut. Projektin aikana opin paljon uutta. Tilaajakin tuntui olleen tyytyväinen aikaansaannokseemme. Vaikka projekti oli pääsääntöisesti mukava, niin dokumenttien laatimisesta sekä väli- ja loppuesittelyistä en juurikaan nauttinut. Vaikka en pidä kirjoittamisesta, enkä esiintymisestä, sain näilläkin osa-alueilla arvokasta kokemusta, josta on vielä tulevaisuudessa varmasti hyötyä. Minulle henkilökohtaisesti tärkeimpänä sovellusprojektissa oli kuitenkin, että se palautti kadoksissa olleen motivaationi alaa kohtaan.

13.5 Heikki Salon kokemuksia omin sanoin

Harkitsin ennen sovellusprojektia pitäväni päivätyöni sovellusprojektin ajan pienemmällä tuntimäärällä taustalla, vaikka tiesin sovellusprojektin olevan työläs. Onneksi jätin työt kokonaan pois kuultuani suosituksia, sillä sovellusprojekti oli vaativampi kuin odotin. Toivoin pääseväni ohjelmointitehtäviin ja pääseväni opiskelemaan uusia tekniikoita. Projektin alussa kuitenkin kannustettiin valitsemaan itselle vieraita tehtäviä, joten lupauduin Verso-projektiin projektipäälliköksi, vaikka tiesin roolin sisältävän paljon dokumentointia ilman lainkaan ohjelmointia.

Arvuuttelin ennen projektia, millainen ryhmä kohdalleni sattuisi. Lähtökohta projektille oli jännittävä, sillä projektipäällikön rooli oli täysin vieras, kun puolestaan edessä oleva ohjelmointi tekniikoineen ja työkaluineen oli muulle ryhmälle ennalta lähes täysin vierasta. Alkuun päästiin kuitenkin nopeasti, sillä ryhmä opiskeli uutta härkämpäisesti ja rimaa pidettiin koko ajan korkealla. Nelihenkinen ryhmä oli viikoilla 11 ja 15 yli 140 tuntia yhteensä projektikopissa, mikä kertoo ryhmän sitoutumisesta tekemiseen.

Kokemukseni projektipäällikkönä ovat kaksinaiset. Toisaalta projektista jäi onnistumisen tunteita esimerkiksi projektiin liittyvien asioiden organisoinnista eri ryhmien välillä. Koen kuitenkin projektipäällikkönä alisuoriutuneeni joiltain tärkeiltäkin osin, sillä projektisuunnitelma ja ajankäyttösuunnitelma viivästyivät pahasti.

Projektissa hankalimmiksi kokemani asiat liittyvät projektikurssin dokumenttikeskisyyteen, sillä jokaisen ilmaistun ajatuksen on löydyttävä asiakirjamuodossa, oikoluettuna ja kauniiksi ladottuna. Dokumenttikeskisyys näkyy myös siten, että sovellusprojektikurssilla keskitytään ohjaamaan asiakirjojen, ei ajatusten kehitystä. Sovellusprojektikurssin sisällössä on aihetta Verso-projektia laajemmallekin keskustelulle.

Täytyy tunnustaa, että ennalta kuulemani perusteella suhteeni dokumentteihin oli kriisiytynyt jo projektin alkaessa Jukka-Pekka Santasen ohjaamana. Projektin lopuksi minun on kuitenkin todettava, että Jukka-Pekan projektin ohessa pitämästä dokumenttikoulusta tulee olemaan minulle varmasti hyötyä jatkossa.

Sovellusprojektikurssista jää ehdottomasti mieluisimpina mieleen hyvä ja toimeentuleva ryhmä sekä mielenkiintoinen aihe. Voisi sanoa, että hienosti toimi kaikki ryhmään, tekniikkaan ja sovellukseen, siis kaikenkaikkiaan tekemiseen liittyvä sovellusprojektikurssilla, jättäen hyvät muistikuvat.

14 Yhteenveto

Verso-projekti toteutti Jyväskylän yliopiston tietotekniikan laitokselle prototyypin lähdekoodien julkistamisjärjestelmästä. YouSource-sovellus kehitettiin Ruby-kielellä Ruby on Rails -sovelluskehystä käyttävän Gitorious-sovelluksen päälle.

Projektin aikana projektiryhmä oppi työskentelemään yhdessä ohjelmistoprojektissa, sekä toimimaan useiden eri sidosryhmien kanssa. Ryhmä sai myös kokemusta useista sovelluskehityksessä tarvittavista työkaluista ja projektin läpiviennistä.

Ryhmätyöskentelyn ja sovelluksen osalta projektia voi pitää onnistuneena. Sen sijaan puutteelliseksi Verso-projekti jäi projektin läpivientiin liittyvien projekti- ja tuntisuunnitelman osalta.

Projekti alkoi 3.2.2010, ja se päättyi projektiraportin valmistuttua 15.12.2010. Projekti myöhästyi suunnitellusta noin puoli vuotta, sillä projektin viimeistelylle ja raporteille ei oltu varattu riittävästi aikaa.

15 Lähteet

- [1] Antti-Juhani Kaijanaho, "Verso-projekti, Teknisen ohjaajan huomioita koodista", saatavissa HTML-muodossa
<URL: <https://korppi.jyu.fi/kotka/servlet/list-archive/verso/0771.html>>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [2] Christian Neukirchen, "Ruby Style Guide", saatavissa HTML-muodossa
<URL: <http://github.com/chneukirchen/styleguide/raw/master/RUBY-STYLE>>, 2010.
- [3] Gitorious, "About Gitorious", saatavissa HTML-muodossa
<URL: <http://gitorious.org/about>>, 2010.
- [4] Gitorious, "Gitorious", saatavissa XHTML-muodossa
<URL: <http://groups.google.com/group/gitorious?pli=1>>, 2010.
- [5] Gitorious, "Gitorious Coding Style Guide", saatavissa HTML-muodossa
<URL: <http://gitorious.org/gitorious/pages/CodingStyleGuide>>, 2010.
- [6] Heikki Salo, "Verso-projekti, 1. koodikatselmoinnin muistio", saatavissa raakatekstimuodossa
<URL: <http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/katselmoinnit/katselmointi1.txt>>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [7] Heikki Salo, "Verso-projekti, Projektisuunnitelma", saatavissa PDF-muodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/projektisuunnitelma/Verso_projektisuunnitelma_0.6.0.pdf>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [8] Heikki Salo, "Verso-projekti, Projektiraportti", saatavissa PDF-muodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/projektiraportti/Verso_projektiraportti_1.0.pdf>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [9] Heikki Salo, "Verso-projekti, Suunnitelma esikäyttäjien hyödyntämisestä", saatavissa raakatekstimuodossa

- <URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/suunnitelma_esikayttajien_hyodyntamisesta.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [10] Heikki Salo, "Verso-projekti, Tarvekuvaus Korppi-ryhmien käyttämisestä", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/tarvekuvaus_korppi-ryhmien_kayttamisesta.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [11] Heikki Salo, "Verso-projekti, YouSource-sovellukseen kirjautuminen KorppiL-DAPilla", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/yousource-sovellukseen_kirjautuminen_korppildapilla.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [12] Juho Nieminen, "Verso-projekti, 1. käytettävyydestä muistio", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/kaytettavyys_dokumentit/kaytettavyystestaus_1.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [13] Juho Nieminen, "Verso-projekti, 1. palaverin pöytäkirja", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/palaverit/poytakirja_1_palaveri.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [14] Juho Nieminen, "Verso-projekti, 2. käytettävyydestä muistio", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/kaytettavyys_dokumentit/kaytettavyystestaus_2.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [15] Juho Nieminen, "Verso-projekti, 2. palaverin pöytäkirja", saatavissa raakatekstimuodossa
<URL: <http://sovellusprojektit.it.jyu.fi/verso/src/>

- docs/project/palaverit/poytakirja_2_palaveri.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [16] Juho Nieminen, "Verso-projekti, Sovellusraportti", saatavissa PDF-muodossa <URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/application/application_report/application_report_1.0.pdf>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [17] Marko Peltola, "Verso-projekti, 2. koodikatselmoinnin muistio", saatavissa raakatekstimuodossa <URL: <http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/katselmoinnit/katselmointi2.txt>>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [18] Marko Peltola, "Verso-projekti, 4. palaverin pöytäkirja", saatavissa raakatekstimuodossa <URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/palaverit/poytakirja_4_palaveri.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [19] Marko Peltola, "Verso-projekti, YouSource RHEL Installation Guide", saatavissa HTML-muodossa <URL: http://yousource.it.jyu.fi/yousource/pages/Installation_Guide>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [20] Petri Heinonen, Ajankäytönseurantasovellus, saatavissa Excel-muodossa <URL: <http://appro.mit.jyu.fi/tools/ajankaytto/ajankaytonseuranta.xls>>, Jyväskylän yliopisto, informaatioteknologian tiedekunta, viitattu 11.11.2010.
- [21] Sam Ruby, Dave Thomas, David Heinemeier Hansson, et al "Agile Web Development with Rails", Pragmatic Bookshelf, 2009.
- [22] Tero Hänninen, "Verso-projekti, Käytettävyysspäivän muistio", saatavissa raakatekstimuodossa <URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/kaytettavyys_dokumentit/kaytettavyyspaiva.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.

- [23] Tero Hänninen, "Verso-projekti, System Test Plan", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/application/system_tests/system_test_plan_21st_May.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [24] Tero Hänninen, "Verso-projekti, System Test Report for Version RC1". saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/application/system_tests/system_test_report-rc1.pdf>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [25] Tero Hänninen, "Verso-projekti, System Test Report for Version RC2", saatavissa PDF-muodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/application/system_tests/system_test_report-rc2.pdf>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [26] Tero Hänninen, "Verso-projekti, Vertailu alustoista", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src/docs/project/vertailu_alustoista.txt>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [27] Ville Tirronen, "Verso-projekti, Alustavaa lukemistoa projektiin liittyen", Jyväskylän yliopisto, tietotekniikan laitos, 2010.
- [28] Wikipedia, "Gitorious", saatavissa HTML-muodossa
<URL: <http://en.wikipedia.org/wiki/Gitorious>>, 2010.
- [29] Verso-projekti, "Usability suggestions to dashboard vs. profile, activity feed and projects", saatavissa HTML-muodossa
<URL: http://groups.google.com/group/gitorious/browse_thread/thread/a6a888ff10b770ef/>.
- [30] Verso-projekti, "Verso-projektin tehtävähallinta", saatavissa HTML-muodossa
<URL: <https://trac.cc.jyu.fi/projects/verso>>, Jyväskylän yliopisto, tietotekniikan laitos, 2010.