Verso-sovellusprojekti

Projektiraportti

Tero Hänninen
Juho Nieminen
Marko Peltola
Heikki Salo

Versio 0.8
Julkinen
11.11.2010

Jyväskylän yliopisto

Tietotekniikan laitos

Jyväskylä

Hyväksyjä	**Päivämäärä**	**Allekirjoitus**	**Nimenselvennys**
Projektipäällikkö	__.__.2010		
Tilaaja	__.__.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: 69

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, tehtävät, testaus, Trac, 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
0.4.0	6.6.2010	Kirjoitettu riskien toteutumista kuvaava luku. Täydennetty tavoitteita kuvaavaa lukua.	HS

Versio	Päivämäärä	Muutokset	Tekijät
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 prosessimallia 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	Santasen esittämiä esitystapakorjauksia lukuihin 1-7.	HS
0.8.0	11.11.2010	Santasen esittämiä esitystapakorjauksia lukuihin 8-14.	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

Tilaaja:

- 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

Verso-projekti oli Jyväskylän yliopiston tietotekniikan laitoksella keväällä 2010 toteutettu sovellusprojekti. Projekti määritteli, suunnitteli, toteutti ja testasi Jyväskylän yliopiston tietotekniikan laitokselle prototyypin lähdekoodien julkistamisjärjestelmästä.

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

Lähdekoodien julkistamista varten Verso-projekti kehitti Gitorious-järjestelmän pohjalta WWW-sovelluksen, jolla käyttäjät voivat luoda lähdekoodit sisältäviä tietovarastoja ja julkistaa niitä. Tietovarastoja voi ylläpitää käyttäen hajautetun versiohallinnan työkaluja.

Toteutetun sovelluksen vaatimuksia ja toteutusratkaisujen kartoitustehtäviä ylläpidettiin projektissa Trac-järjestelmässä [29]. Sovelluksen rakenne ja toiminta kuvataan sovellusraportissa [22]. Projektisuunnitelma [18] määritteli osittain projektin läpiviennin käytänteet, resurssit ja aikataulun. Dokumentissa arvioidaan projektisuunnitelman toteutumista.

Luvussa 2 kuvataan projektiin liittyviä termejä. Luku 3 tarkastelee projektin taustaa sekä projektin 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 tehtävien ja työmäärän jakautumista projektin jäsenten kesken. Luvussa 11 kuvataan Verso-projektissa käytettyä prosessimallia ja projektin aikaväliä. Luku 12 kuvaa riskien toteutumista. Luvussa 13 ryhmän jäsenet kuvaavat kokemuksiaan projektista.

2 Termejä

Luvussa kuvataan dokumentissa esiintyviä termejä aihealueittain.

2.1 Projektin läpivientiin liittyviä termejä

Luvussa kuvataan projektin läpivientiin liittyviä termejä. Projektissa käytettiin Trac-projektinhallintaohjelmistoa, jonka käyttöä on kuvattu tarkemmin luvussa 9.

Ketterä ohjelmistokehitys on joukko käytettäviä menetelmistöjä, joille on yhteistä toimivan ohjelmiston ensisijaisuus, suora viestintä ja nopea muutoksiin reagointi projektin aikana.

Tiketti (engl. *ticket*) on yksittäinen ominaisuusvaatimus tai muu tehtäväkuvaus.

Trac on projektinhallintaohjelmisto ja tikettijärjestelmä, jolla voi muun muassa hallita projektin vaatimusmäärittelyä.

Vaihe (engl. *phase*) on lyhyt ajanjakso projektin toteutuksessa.

2.2 Git-versiohallintaohjelmiston termejä

Luvussa kuvataan projektin tulosten hallintaan ja sovelluksen kehittämisessä käytetylle Git-versiohallintaohjelmistolle ominaisia termejä. Versiohallinnan käytöstä lisää luvussa 8.

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.

Repository on englanninkielinen vastine termille tietovarasto.

Tietovarasto (engl. *repository*) on versioitu tiedon tallennustapa. Dokumentissa viitataan yleensä nimenomaan Git-ohjelmistolla ylläpidettyyn tietovarastoon.

2.3 Sovellukseen liittyviä termejä

Luvussa kuvataan projektissa valmistatetulle ohjelmistolle ja sen pohjana käytetylle Gitorioukselle ominaisia termejä, ja täydentää Git-versiohallinnan termistöä sovelluksen osalta.

Gitorious on WWW-sovellus Git-tietovarastojen selaamiseen ja hallintaan. item[Julkistaminen] tarkoittaa lähdekoodin 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. *daemon*).

Projekti on YouSource-järjestelmässä käyttäjän luoma sisältökokonaisuus, jolla voi olla monia tietovarastoja.

Wiki	on verkkosivukokonaisuus, jonka tavoitteena on helppo muokattavuus ja sivujen välinen vuorovaikutteisuus.

3 Tavoitteiden toteutuminen

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

3.1 Taustaa

Jyväskylän yliopistossa toimii monenlaisia tutkimusryhmiä, joissa tuotetaan 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 edes kehityksen aikana ryhmän sisällä, mikä saattaa johtaa päällekkäisen työn tekemiseen. Työntekijän lähtiessä talosta hänen yksin valmistamansa lähdekoodi käytännössä menetetään, kuten tilaajan alustavassa aihekuvauksessa [7] kuvaataan.

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 [4]. Lähdekoodin jakamisen ollessa työlästä, on riskinä, ettei lähdekoodi leviä edes saman tutkimusryhmän sisällä.

Yhtenäisen tallennuspaikan puuttumisen ohella lähdekoodien hyödyntämiseen liittyy oikeuksiin liittyviä haasteita. Käytäntöjen puuttuessa lähdekoodin yhteyteen ei nimittäin aina tallenneta tietoa sen lisenssistä. Niinpä tutkimusryhmien osaamista on vaikea markkinoida jakamalla lähdekoodia, kun ei ole tietoa, onko lähdekoodin lisenssi yhteensopiva edes 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

Tilaajana 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.

Tärkeimmät tilaajan tavoitteet kohdistuivat käytön helppoon aloittamiseen ja yksityisten tietovarastojen mahdollistamiseen. Projektin kuluessa järjestelmälle määrittyi monia pienemmälle prioriteetille määriteltyjä vaatimuksia.

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

Taulukko 3.1: Toteutettujen ja toteuttamattomien 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:

- Korppi-opintotietojärjestelmän käyttäjätunnuksillä kirjautuminen

- projektien ja tietovaraston näkyvyyksien asteittainen rajoittaminen
- ohjattu uuden tietovaraston luonti
- selausnäkymä tietovarastoille
- tietovaraston metatietojen tallentaminen tietovarastoon
- tietovaraston päivittäminen tiedoston tai paketin antamalla
- sovelluksen ajossa olevan version tunnisteen näyttäminen käyttäjille
- tietovaraston asettaminen peilaamaan ulkoista SVN- tai tiedostopaketti- tietovarastoa
- lisenssien valitseminen tietovarastokohtaisesti projektikohtaisuuden sijaan sekä
- projektin wikin käyttämisen mahdollistaminen projektin muiden tietovarastojen tapaan sekä
- yleisiä käytettävyysparannuksia.

Tilaajan kanssa siirrettiin jatkokehitykseen tavoitteina olleista ominaisuuksista seuraavat:

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

Toteutettu järjestelmä kattaa tärkeimmät siihen kohdistuneet vaatimukset. Projektin tuloksena on kehityskelpoinen järjestelmä, jota voi jo sellaisenaan käyttää lähdekoodien julkistamiseen ja työkaluna ohjelmoinnin aikana. Sovelluksen toteuttamista kuvataan tarkemmin sovellusraportissa [22].

Käyttöliittymän osalta Gitorioukseen ei tehty merkittäviä rakenteellisia muutoksia, mutta projektin aikana käyttöliittymiin tehtiin useita projektin aikana ilmenneisiin parannusehdotuksiin perustuvia käytettävyyskorjauksia. Uusien ominaisuuksien myötä Gitorioukseen lisättiin myös muutama täysin uusi käyttöliittymä.

3.3 Tavoitteiden muutokset projektin kuluessa

Suurin osa tärkeimmistä vaatimuksista oli selvillä projektin alusta asti. Esimerkiksi 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 Gitoriouksessa ovat.

Käytön aloittamisen helpottamiseksi muun muassa tietovaraston luontia Gitoriousovelluksessa 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.

Verso-projekti toteutti YouSource-järjestelmään kaikki pakollisiksi määritellyt ja lähes kaikki tärkeiksi määritellyt ominaisuudet. Suurin osa jatkokehitykseen jätetyistä ominaisuuksista karsittiin 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 [29] ja sovellusraporttiin [22]. Trac-tiketit toimivat mahdollisena pohjana jatkokehitykselle. Projektiryhmästä Marko ja Tero jatkavat Verso-projetin loputtua jatkokehittämään sovellusta tietotekniikan laitokselle.

3.5 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, minkä lisäksi Juho, Tero ja Marko halusivat projektin aikana saada lisää kokemusta ohjelmoinnista sekä oikeassa ohjelmistoprojektissa mukanaolosta ja esimerkiksi versiohallinnan käyttämisestä. Heikki halusi saada kokemusta projektin läpiviennistä, minkä johdosta hänet valittiin projektipäälliköksi. Kaikki projektiryhmän Verso-projektille asettamat oppimioppimistavoitteet täyttyivät.

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

3.6 Projektin kuluessa opituista tekniikoista

Verso-projektin jäsenet saivat Gitorious-sovellusta jatkokehittäessään kokemusta Ruby-ohjelmointikielestä, Ruby on Rails -ohjelmistokehyksestä sekä WWW-sovelluksen ylläpitämisestä. Tietokantana toimi MySQL 5.0 -tietokannan hallintajärjestelmä, ja projektin aikana ryhmäläiset oppivat muun muassa SQL-kieltä.

Projektiryhmä oppi projektin aikana käyttämään hajautettua versiohallintaa käyttämällä Git-versiohallintaohjelmistoa sekä sovelluksen kehittämiseen että projektin dokumenttien hallintaan. Ryhmä oppi myös avoimen lähdekoodin ohjelmistopro-

jekteihin liittyviä versiohallintakäytänteitä. Lisäksi oman sovelluksen käyttämiselle ja esikäyttöön tarjoamiselle projektin aikana edellytys oli, että ryhmä oppi kehittämään keskeneräisiä ominaisuuksia omissa haaroissaan versiohallinnassa, jotta 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 tietotekniikan laitoksen ATK-käytänteiden kanssa halutessaan muutoksia työpöytäkoneille esimerkiksi asennettavien sovellusten osalta.

3.7 Projektiin liittyvät dokumentit

Projektin kuluessa valmistetut tulokset voidaan jakaa sovellukseen sekä projektiin liittyviin tuloksiin.

Projektin tulokset on jaoteltu 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 lisäksi seuraavat julkiset suunnitelmat ja raportit:

Ajankäyttöraportti sisältää ryhmän jäsenten kirjaamat työtunnit.

Asennusohje sisältää sovelluksen asennusohjeet.

Esikäyttäjien hyödyntämissuunnitelma kuvaa esikäyttäjien hyödyntämistä projektissa sekä kuvaa käytettävyystestauksien läpivientiä.

Esittelymateriaalit	sisältävät väli- ja loppuesittelyn esitysgrafiikan ja muistiot.
Järjestelmätestaussuunnitelma	määrittelee sovelluksen toimivaksi toteamiseen vaaditut kohdat.
Järjestelmätestausraportit	kuvaavat kehitetyn sovelluksen järjestelmätestauksen tulokset.
Käytettävyysmuistiot	sisältävät kahden käytettävyystestauksen muistiot.
Projektiraportti	kuvaa projektin läpiviennin ja asetettujen tavoitteiden saavuttamista.
Projektisuunnitelma	kuvaa projektin tavoitteita, tehtäviä, aikataulua, yleisiä käytäntöjä ja riskien hallintaa.
Sovellusraportti	kuvaa toteutetun sovelluksen 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, ja siinä oli vasta alustavasti kuvattu riskejä sekä tuntien jakautumista.

Edellä mainittujen dokumenttien lisäksi ryhmä laati seuraavat tulokset:

Alustavertailu	sisältää projektin alussa valmistetun vertailun potentiaalisista alustavaihtoehdoista.
Itsearvioinnit	sisältävät ryhmän jäsenten arvioinnit omasta panoksesta, onnistumisesta ja oppimisesta.
KorppiLDAPin käyttökuvaus	kuvaa, miten projektissa valmistettu sovellus käyttää KorppiLDAPia.

Lisenssisopimus	sisältää jäsenten suostumuksen projektissa tuotetun ohjelmakoodin sijoittamisesta AGPLv3-lisenssin alaisuuteen.
Lähdekoodi	sisältää valmistetun lähdekoodin kommentteineen.
Palaverien dokumentit	sisältävät palaverien esityslistat ja pöytäkirjat.
Palavereiden esityskalvot	sisältävät palavereissa käsitellyt asiat tilakatsauksineen 4. palaverista alkaen.
Sähköpostiarkistot	sisältävät kaikki projektin sähköpostilistalla käydyt keskustelut, mukaanlukien esimerkiksi päivittäiset tilakatsaukset.

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 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 on 3. vuoden ja Marko Peltola 7. vuoden tietotekniikan opiskelija. Juho Nieminen ja Heikki Salo ovat 4. vuoden tietotekniikan opiskelijoita. Opiskelijoista Heikillä oli kokemusta WWW-sovellusten kehittämisestä sekä ohjelmistoprojektissa toimimisesta Korppi-kehityksessä toimimisen kautta. Ryhmällä ei ennen projektia ollut kokemusta projektin läpiviennistä, eikä suuresta osasta kehitysympäristöä. Kaikilla ryhmän jäsenistä oli vahvuutena hyvät sosiaaliset taidot sekä erinomaiset valmiudet omaksua uusia työkaluja ja tekniikoita.

Tilaajana 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 [6]. 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 suoritetun puhe- ja kirjoitusviestinnän kurssin osalta projektiryhmää auttoivat kurssin opettajat Leena Peltomaa (kirjoitusviestintä) sekä Minna Koljonen (puheviestintä). Käytettävyysasioissa projektiryhmää auttoi Meeri Mäntylä, joka piti ryhmälle käytettävyyspäivän.

Graafinen suunnittelija Auri Kaihlavirta laati sovellukselle logokuvan ja määritti

värit, joiden avulla sovellukselle saatiin persoonallinen ilme.

Projektissa toteutettua sovellusta tarjottiin esikäyttöön, ja projekti tekikin kevään aikana yhteistyötä parinkymmenen tietotekniikan laitksen 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.

Projektiryhmän käytössä oli lisäksi sovellusprojektien kannettava tietokone, videoprojektori 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.

Ryhmän käytössä oleviin Fedora Core -tietokoneisiin oli ATK-lähituen toimesta asennettu palvelinohjelmistoina Apache, Git, MySQL sekä Ruby ja Ruby on Rails -ohjelmakirjastot, jotka mahdollistivat Gitorious-sovelluksen kehittämisen. Windows XP -tietokoneeeseen oli ATK-tuelta pyydetty TortoiseGit- ohjelmisto versiohallintaa ja Microsoft Office 2003 ajankäytön merkitsemistä varten.

Linux-tietokoneisiin oli projektia varten asennettu MySQL 5.0 -tietokantaohjelmisto, Apache-WWW-palvelin, Ruby-ohjelmatulkki sekä Ruby on Rails -sovelluskirjasto.

Projektiryhmällä on lisäksi verkon kautta käyttöoikeus myös erilliseen testipalvelimeen `versotest.it.jyu.fi`, jonka käyttöjärjestelmänä oli Red Hat Enterprise Linux 5.0.

Projektiryhmä ei käyttänyt ohjelmointiin raskaita ohjelmointityökaluja- tai ympäristöjä. Ohjelmointi sekä dokumentointi suoritettiin pääasiassa käyttäen Notepad++-ohjelmaa (Windows XP -työasema) sekä Gedit- ja Vim-ohjelmia (Fedora Core -työasemat).

Työmäärien kirjaamiseen käytettiin Petri Heinosen kehittämää Excel-pohjaista ajankäytönseurantasovellusta.

Projektiorganisaation 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

Projekti vastasi luentojen ja perehdytysten osalta suunniteltua. Projektissa vaadittiin hyvää tuntemusta Git-versiohallinnasta, joka ei ollut ryhmäläisille 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.

Ohjelmointikielenä oli Ruby ja alustana Ruby on Rails, jotka eivät olleet ryhmän jäsenille ennestään tuttuja. Ryhmä tutki kirjallisuutta verkosta ja tilasi Santasen kautta käyttöönsä Agile Web Development with Rails (Ruby, Thomas, Heinemeier Hansson) -kirjan.

Projektissa toteutettun prototyypin ohjelmointiin tarvittiin tuntemusta Rubystä, Ruby on Railsista sekä sovelluksen pohjana toimivan Gitoriouksen toiminnasta. Projektiryhmä perehtyi Rubyyn ja Ruby on Railsiin omatoimisesti tutkimalla Internetissä saatavilla olevia ilmaisia oppaita. Gitoriouksen asentamiseen ja toimintaan projektiryhmä tutustui lähdekoodeihin perehtymällä sekä kysymällä apua Gitoriouksen kehitysyhteisöltä sähköpostilla ja IRC-kanavalla.

Jukka-Pekka Santanen piti luennon projektin hallinnasta ja läpiviennistä. Luennolla käsiteltiin myös ryhmän jäsenten keskinäistä viestintää.

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

- esittely ja esiintyminen (Koljonen)
- kirjoitusviestintä (Peltomaa)
- kokous- ja neuvottelukäytänteet (Santanen)
- 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 opetussisältöön sisältyi myös katsaus sovellusten käytettävyyteen. Meeri Mäntylä piti projektiryhmälle käytettävyyspäivän, jossa käytiin läpi yleisiä käytettävyyteen liittyviä seikkoja sekä harjoiteltiin käytettävyystestauksen pitämistä. Lopuksi Mäntylä koekäytti sovellusta ja kertoi huomioitaan, joista kirjattiin muistio [12].

5 Yhteistyö Gitorious-yhteisön kanssa

Projektisuunnitelma [18] rajoittui kuvaamaan Gitorious-yhteisön kanssa toimimista vain ohjelmiston osalta. Projektin tavoitteena oli valmistaa alkuperäisen Gitoriouksen kanssa yhteensopivia ominaisuuksia, joita yritettäisiin saada alkuperäiseen Gitorious-ohjelmistoon ja/tai käyttöön muille kehittäjille.

Suunnitelmasta poikettiin teknisten ongelmien vuoksi, jotka hankaloittivat ominaisuuksien saamista käyttöön YouSourcen ulkopuolella. Julkaisua on kuvattu luvussa 5.2.

Projektiryhmä kommunikoi projektin aikana Gitorious-yhteisön kanssa pääasiassa Freenode-verkossa #gitorious-IRC-kanavan sekä Gitorious-sähköpostilistan [10] avulla.

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 Gitorioukseen pohjautuessaan jo lisenssinsä puolesta avointa lähdekoodia, mutta avoimuuden lisäksi Gitoriouksen päälle rakennettavat muutokset pidettiin mahdollisimman yhteensopivina alkuperäisen järjestelmän kanssa. Alkuperäisen järjestelmän henkeen toteutetuilla ja yhteensopivilla kompo-

nenteilla on parhaat odotukset päätyä osaksi alkuperäistä järjestelmää, Gitoriousmainlineä.

Alkuperäisen ohjelmiston osaksi saadut mahdolliset muokkaukset hyödyttävät kaikkia osapuolia. Etenkin Verson tuottamien ominaisuuksien saamisesta Gitorioukseen hyötyy ne kehittänyt tilaaja, sillä siten ominaisuuksien ylläpito ei ole enää yksin alkuperäisen kehittäjän vastuulla. Alun perinkinhan 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 buginkorjauksia lähetettiin muun yhteisön saataville vasta keväällä, 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 mainlineversion ulkoasun vaihduttua 18.5.2010, kun projektilaiset esittivät sekä IRC-kanavalle että postilistalle parannusehdotuksia siihen. Ulkoasun kehittänyt Ole Martin Kristiansen kiitti ehdotuksista ja kertoi toteuttaneensa yhden ja kirjanneen loput kehitettäväksi [28].

6 Käytänteet

Luvussa kuvataan projektin suunniteltuja käytettyjä käytänteitä ja niiden toteutumista. Käytänteet pyrkivät ennenkaikkea varmistamaan tulosten ja projektin laatua. Sovellukseen keskittyvää laadunvarmistusta on kuvattu myöhemmin luvussa 7. Versiohallintaan liittyvät käytänteet ovat laajuutensa takia kuvattu omassa luvussa 8.

Projektisuunnitelmassa [18] kuvatut käytänteet toteutuivat lähes sellaisinaan, eikä niistä poikettu tietoisesti. Suurin poikkeama ovat 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 käyttönotto.

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`. YouSource-sovelluksen esikäyttäjille 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 ongelmista 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 palaverikäytänteet 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 [4] 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 sihteeriä. Projetiryhmä kierrätti puheenjohtajan ja sihteerin rooleja siten, että kukin toimi projektin aikana vähintään kahdesti puheenjohtajana ja sihteerinä. Palavereissa assiat 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 tarksitettavaksi etukäteen, joten palavereissa voitiin käydä läpi edellinen palaveri vain päätöksien 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 projektiryhmän tarkastettavana, minkä jälkeen sihteeri lähetetti 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 alussa sovittiin[4].

Tilaajan 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 [4].

6.4 Julkistettujen tulosten versionumerointi

Dokumenteissa käytettiin kolmitasoista versionumerointia ensimmäisen palaverin [4] mukaisesti. Ensimmäinen ryhmän jäsenten ulkopuolelle julkistettu versio oli 0.1. Ryhmän sisäiseen käyttöön tarkoitettuja versionumeroita kasvatettiin 0.0.1:llä, 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, ja sovelluksen julkistamishetkellä versionumero olisi 1.0 ja kasvaisi uuden ominaisuuden tullen yhdellä [4]. 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 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

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

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

Sovellukseen liittyvä dokumentaatio on englanniksi, projektiin liittyvä suomeksi. Allaolevasta listasta 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.

6.6 Tulosten koostaminen

Verso-projektin kehitystyö sekä projektitiedostojen 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 tallennetaan lisäksi CD:lle**, josta toimitetaan kappaleet tilaajalle sekä projektikansioon. Sähköinen materiaali pitää sisällään luvussa 3.7 mainittujen tulosten tietovarastojen uusimmat versiot, tietovarastojen versiohistorian, Trac-järjestelmän tietoineen ja sähköpostiarkistot. CD:ltä löytyvä sähköinen materiaali on saatavilla myös Verso-projektin `http://sovellusprojekt`
sivuston kautta.

Projektin tuloksista on koostettu **kansio**, johon on tulostettu lisäksi projektin sähköpostiarkistot. Lisäksi kansiossa on kappale allekirjoitetusta **lisenssisopimuksesta**, projektin väliesittelyistä saadut palautteet, jäsenten itsearvioinnit sekä projektin arviointiin sekä vastaavan ohjaajan että tilaajan lausunnot.

7 Sovellukseen liittyvä laadunvarmistus

Projektiin ei osin sen prototyyppiluonteen vuoksi ja osin kesken jääneen projektisuunnitelman vuoksi suunniteltu kattavaa testausta. Tilaaja vaati projektin alusta lähtien esikäyttäjien käyttämistä, mutta toisaalta esimerkiksi järjestelmätestaus suunniteltiin täysin projektisuunnitelman [18] laatimisen keskeyttämisen jälkeen. Luvussa kuvataan sovelluksen laadun varmistamiseen käytettyjä tapoja.

7.1 Lähdekoodin käytänteet

Lähdekoodin käytänteistä sovittiin ensimmäisessä palaverissa [4], että projekti valmistaa käytetyn alustan suosituksia ja käytänteitä noudattavaa lähdekoodia. Hyvää ohjelmointityyliä tavoitellessaan ryhmän oli siis sekä seurattava Ruby-ohjelmointikielen [15] että Ruby on Rails -kehyksen suosituksia, minkä lisäksi Gitorious-projekti määritteli joitain omia ohjelmointikäytänteitä [11].

Ruby-kielinen lähdekoodiesimerkki hyvää ohjelmointityyliä noudattavasta luokasta ja metodimäärittelystä:

```
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
```

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

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

Kehityksessä käytettyjä käytänteitä kuvataan tarkemmin sovellusraportissa [22].

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 korkeintaan kaksi.

Käytännössä projektissa ohjelmoitiin parityönä vaativat ja paljon tutkimista vaativat kokonaisuudet. Parin käyttö haastavissa tilanteissa vaikutti auttavan ratkaisutapojen hahmottelemisessa huomattavasti, minkä 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 Integraatiotestaus

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 muut osioit, joihin kehittäjä arveli ominaisuuden aiheuttaman muutoksen vaikuttavan. Ominaisuutta käyttöönottaessa ei tehty raskasta järjestelmätestausta, vaan kehittäjän oman testauksen jälkeen koodi siirtyi projektiorganisaation ja esikäyttäjien käyttöön.

7.4 Automaattiset poikkeustiedotteet

Projektissa ei alunperin suunniteltu käytettävän automaattisia poikkeustiedotteita, vaan ne olivat myöhemmin löydetty ja käyttöönotettu 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 nojannu ilmoitusahkeruuteen, vaan virhetilanteesta ja siihen johtaneesta sivupyynnöstä saatiin aina tieto.

Poikkeuksen tapahduttua ryhmä päätti keskenään, 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 [26]. Esikäyttäjien hyödyntäminen vastasi suunniteltua.

Tilaajan halusi ottaa 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- sekä henkilökun-

tatiloihin. Tämä esti käytännössä täysin opiskelijoiden hyödyntämisen esikäytössä. Sovellus avattiin koko Jyväskylän yliopiston verkkoon 13.4.2010, joka tapahtui projektiorganisaation oman käytön alkamisesta noin 7 viikkoa myöhemmin. Avaus mahdollisti viimein opiskelijoille sekä kotoaan esikäyttäville 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 esikäyttäjä ei projektin aikana voinut tehdä.

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

7.6 Käytettävyyspäivä

Meeri Mäntylä piti projektiryhmälle käytettävyysluennon 29.3.2010, jossa käsiteltiin sovelluksen käytettävyyteen liittyviä kriteerejä. Lisäksi ryhmä toteutetti ryhmätyönä WWW-sivuston käytettävyystestauksen, mikä toimi hyvänä harjoituksena myöhemmin toteutetuille käytettävyystestauksille. Käytettävyyspäivä toteutui suunnitellusti.

Luennon lisäksi Mäntylä kävi läpi YouSource-sovellusta ja nosti esiin havaitsemiaan seikkoja. Selkeitä käytettävyyspuutteita kirjattiin tiketeiksi, minkä lisäksi käytettävyyspäivästä tehtiin kokonaisuudessaan muistio [12].

7.7 Käytettävyystestaukset

Verso-projekti suoritti YouSource-sovelluksen käytön aloittamiseen liittyvistä toimista kaksi käytettävyystestausta, joissa käytettiin tilaajan nimeämiä testihenkilöitä. Käytettävyystestauksissa esille nousseet ongelmat ja huomiot kirjattiin muistioiksi ja tiketeiksi, minkä lisäksi niiden yleiset huomiot käytiin läpi projektipalaverissa. Ensimmäisestä [13] ja toisesta [14] käytettävyystestauksesta laaditut muistiot löytyvät projektin dokumentaatiosta.

7.8 Koodikatselmoinnit

Sovellusprojekti-kurssiin kuuluu luonnostaan sovelluksen koodikatselmointeja. 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 koodinkatselmoinneissa nostamaan huomioissaan esille myös monia kielen ja sovelluskehyksen käytänteisiin liittyviä seikkoja.

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

Sekä ensimmäisestä että toisesta katselmoinnista ryhmä laati muistiot [2] ja [3]. Kaijanaho lähetti lisäksi 2. katselmoinnin jälkeen lähdekoodista huomionsa [27] projektin sähköpostilistalle.

7.9 Järjestelmätestaukset

Kuten luvun johdannossa mainittiin, järjestelmätestaus suunniteltiin ja toteutettiin täysin projektisuunnitelman [18] valmistamisen keskeyttämisen jälkeen, eikä järjestelmätestausta siis suunniteltu projektin alussa suoritettavan.

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

YouSource-sovelluksen version `verso-project-rc1` järjestelmätestauksessa saatiin kiinni yksi ohjelmointivirhe, joka korjattiin sovellukseen ennen sen lopullista versiota.

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

8 Versiohallinnan käyttö

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

Kesken jääneessä projektisuunnitelmassa kuvattiin projektissa käytetyt versiohallintakäytänteet, 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, on todettava, että oppimisen kannalta Verso-projekti oli erittäin hyödyllinen. Myös havaittujen ongelmien korjaaminen on mahdollista ja pakottaa opettelemaan edelleen uusia menetelmiä.

8.1 Suunniteltu haarojenkäyttöstrategia

Projektin tulokset tallennettiin **Git-versiohallintaan** kahteen eri tietovarastoon. Sovellusta kehitettiin omaan tietovarastoonsa sekä projektin dokumentteja, kuten muistioita, palaverien pöytäkirjoja ja väliesittelymateriaaleja omaan tietovarastoonsa. 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 yhtä aikaa julkistamishaara esikäyttöä varten. Epävakaa ohjelmakoodi oli pidettävä omissa haaroissaan sen koekäyttöön valmistumiseen asti. Kun sovellukseen liittyvä ominaisuus oli varmistettu integraatiotestauksella kehittäjän omalla kehityspalvelimella, sen haara yhdistettiin (engl. *merge*) päähaaraan (engl. *master branch*) ja ominaisuuden historian sisältävä haara jätetään 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 tehdessään yhdis-

tämispyyntöjä (engl. *merge request*) projektiryhmä huomasi luvussa 8.2 kuvattavia puutteita haarojenkäyttöstrategiassaan, jotka tekivät muokkauspyyntöjen lähettämisestä alkuperäiseen 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 omia muutoksiaan 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 kehitys **täysin koskemattomaan Gitorious-sovelluksen päähaaraan**, jonka päälle muutokset olisi pohjattu. 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 haarassa, jota pidetään yhteensopivana Gitorious-kantaversion päähaaran kanssa.

Lisäksi projektin alussa **yhdistämispyyntöjen tekemiseen ei perehdytty käytännössä**. Sen tehdessä 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 kymmeniin. Sopivalla nimeämiskäytännöllä tämä ei kuitenkaan olisi ollut ongelma. Haarojen tekeminen ei varsinaisesti ole yleinen vaatimus, mutta Gitorious-sovelluksessa yhdistämispyyntöjen tekeminen olisi siten helpointa, koska yhdistämispyyntötoiminto ei mahdollista vain tiettyjen muokkauksien (engl. *commit*) poimimista (engl. *cherry pick*).

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

mistuu, siihen liittyvät muokkaukset pitäisi lähettää versiohallintaan vasta, kun toimintoa varten kehitetyt toiminnallisuudet olisi kehitetty ensin ja lähettää työn alla ollut toiminnallisuus vasta sitten.

Selkeiden muutoksien lähettämisestä on kuitenkin 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 synkronointia, mikä voi olla työlästä. 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 YouSource-tietovaraston jälkikäteen tehtyä rakenteen muokkausta, jota on kuvattu tarkemmin luvussa 8.4. Toiseksi, käytänne tikettinumeron lisäämisestä olisi osaltaan auttanut varmistamaan, että muokattavasta asiasta on tehty Trac-tiketti. Tämä käytänne 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 alkupisteesta (engl. *rebase*), jotta kehityshaaraan ei päätyisi ylimääräisiä muutoksia.

8.3 Soveltuvampi haarojenkäyttöstrategia

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

Tärkeintä olisi **koota erillisistä haaroista koostuva oma sovellus omaan haaraansa** 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 ylläpidettävissa 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 muutospyynnöistä 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 [18] 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 kirjoitusasu ja otsikot

Tracia ei käytetty koko 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 käsiteltyä asiaa kyllin selkeästi, kaikilla projektiorganisaatioon kuuluvilla oli mahdollisuus muokata sitä paremmaksi. Tikettien vanha otsikko jäi tikettiin talteen, joten vaatimuuksiin liittyvällä informaatiolla ei ollut mahdollisuutta hukkua muutoksissa.

Otsikkoja 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 palavereissa käsitellyistä asioista, ja ne tarkastettiin ennen toteutusvaiheiden aloittamista palavereissa.

9.3 Sähköposti ja tiketit

Trac-tiketteihin merkittiin muutosilmoitusten vastaanottajaksi cc-kenttään vastaanottajaksi 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, sillä muutosilmoitus lähti kaikista muutoksista. 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**.

9.4 Tikettityypit

Projektisuunnitelmassa [18] kuvatut tikettityypit osoittautuivat käyttökelpoisiksi, ja projekti toteutui pitkälti niitä käyttäen. Trac-sovellus asetti kuitenkin joidenkin puutteidensa osalta haasteita tikettien ylläpitoon, kuten kappaleessa 9.3 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 välittemiseksi.

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

Projektisuunnitelmassa [18] kuvatut tikettien prioriteetit olivat kaikki käytössä, ja projekti toteutui ilman muutoksia niihin.

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.

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

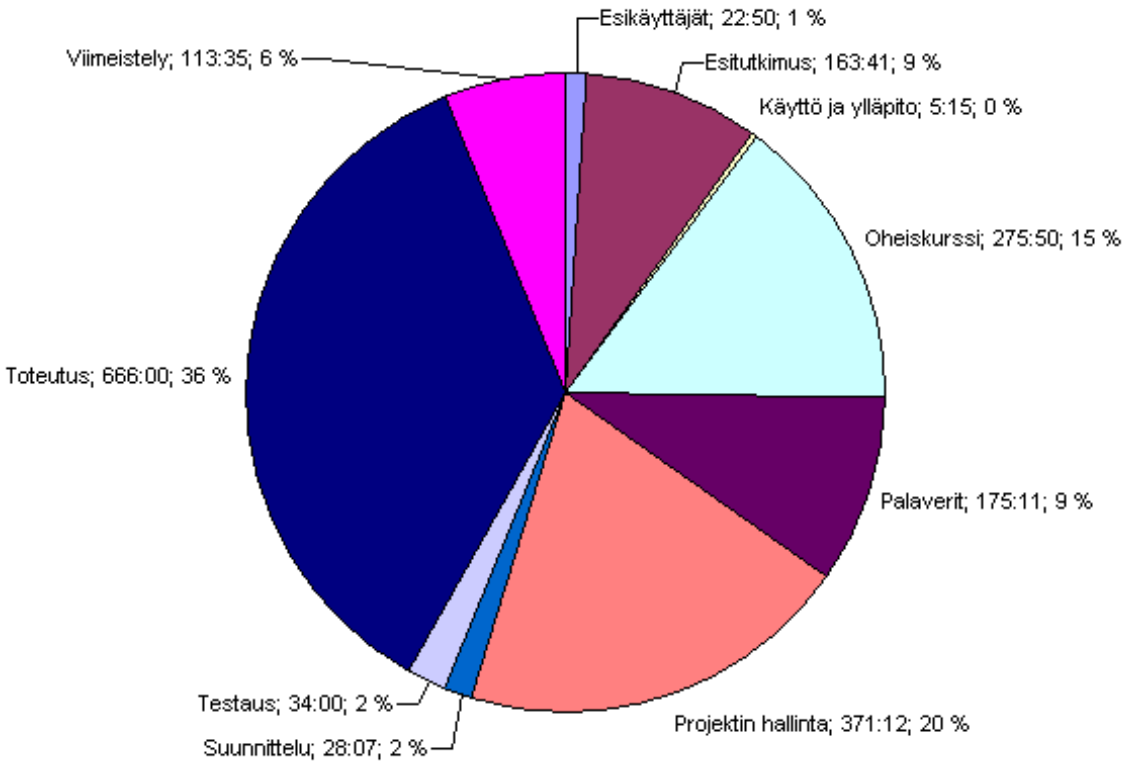
10 Tehtävien työmäärät ja työnjako

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

10.1 Ryhmän ajankäyttö tehtäväkokonaisuuksittain

Verso-projektin ajankäytön jakautuminen tehtäväkokonaisuuksittain on esitetty kuvassa 10.1. Verso-projektilta tilattiin prototyyppi, minkä johdosta ohjelmiston valmistamiseen valittiin projektissa toteutuskeskeinen prosessimalli. Testaukseen (2 %) ja suunnitteluun (2 %) merkittyjen tuntien pieni määrä johtuukirjauskäytänteistä. Vaatimusten toteuttamisesta valitun alustan päälle laadittiin suunnitelmia pääasiassa lähdekoodia lukien ja keskustellessa samalla kokeillen. Myös ohjelmointi ja testaaminen menivät kehitystyössä käsikkäin.

Oheiskurssiin (15 %) on merkitty Verso-projektin ohessa järjestetyille oheiskursseille kuuluvat luennot sekä ohjaajan edellyttämien korjausten tekeminen projektissa valmistettuihin dokumentteihin. Kuitenkin, projektisuunnitelmaa kirjoittaessa tunteja ei vielä krijattu oheiskurssille, joten projektisuunnitelman korjausten tekemiseen kulunut aika puuttuu oheiskursseille kirjatuista.



Kuva 10.1: Ryhmän ajankäyttö tehtäväkokonaisuuksittain

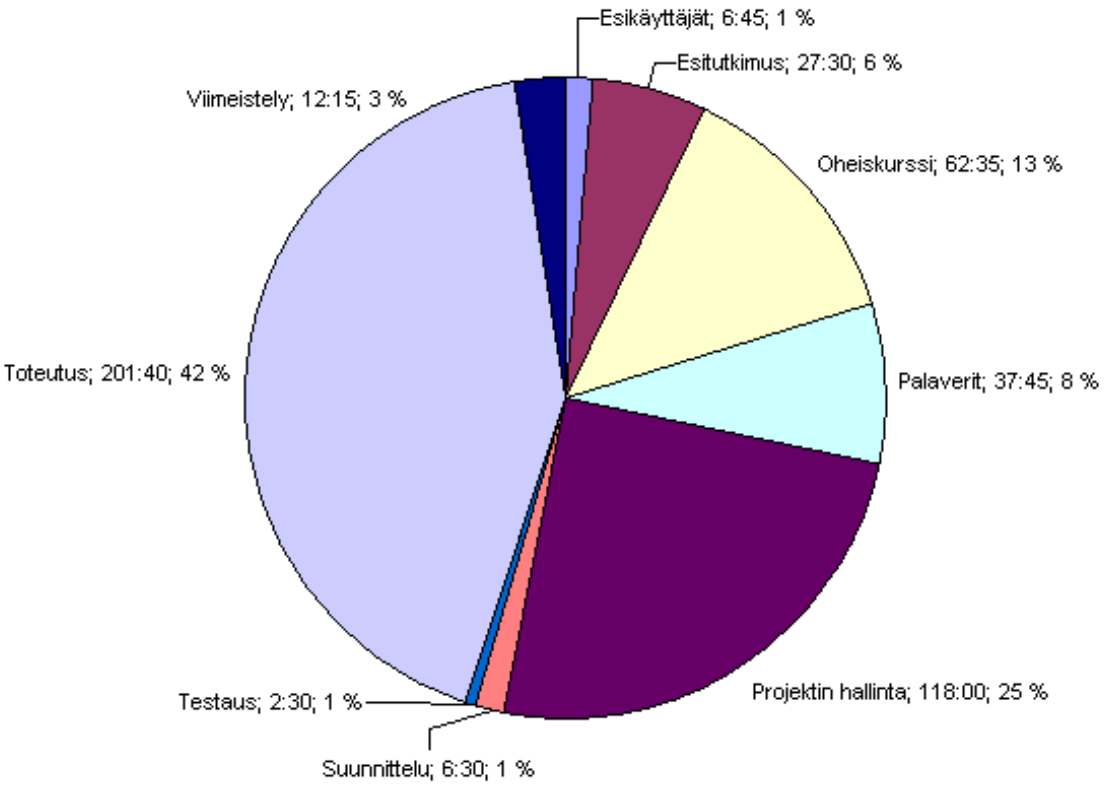
10.2 Juho Niemisen ajankäyttö tehtäväkokonaisuuksittain

Juho toimi Verso-projektin varaprojektipäällikkönä, mikä kuvassa 10.2 näkyy hänen Teroa ja Markoa suurempana projektinhallintaan käytettynä aikana. Juhon vastuulla oli sovellusraportti [22] sekä projektin alussa aloitettu vaatimusmäärittelydokumentti, jonka sisältö kuitenkin myöhemmin siirrettiin Trac-järjestelmään. Juho valmisti palaveridokumenttien lisäksi käytettävyystestauksien muistiot [13] 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.

Kukin jäsenistä toimi projektin aikana projektipalavereissa kahdesti sihteerinä sekä puheenjohtajana. Palavereihin on kirjattu palavereihin osallistumiseen käytet-

ty aika, sisältäen palaverien esityskalvojen tekemisen, palaveriin osallistumisen sekä sihteerin palaverista laatiman pöytäkirjan ensimmäiseen versioon kulunut aika. Pöytäkirjoihin sen jälkeen ohjaajien palautteen perusteella toteutetut muokkaukset on sen sijaan kirjattu tehtäväkokonaisuuden *Oheiskurssi* alle.



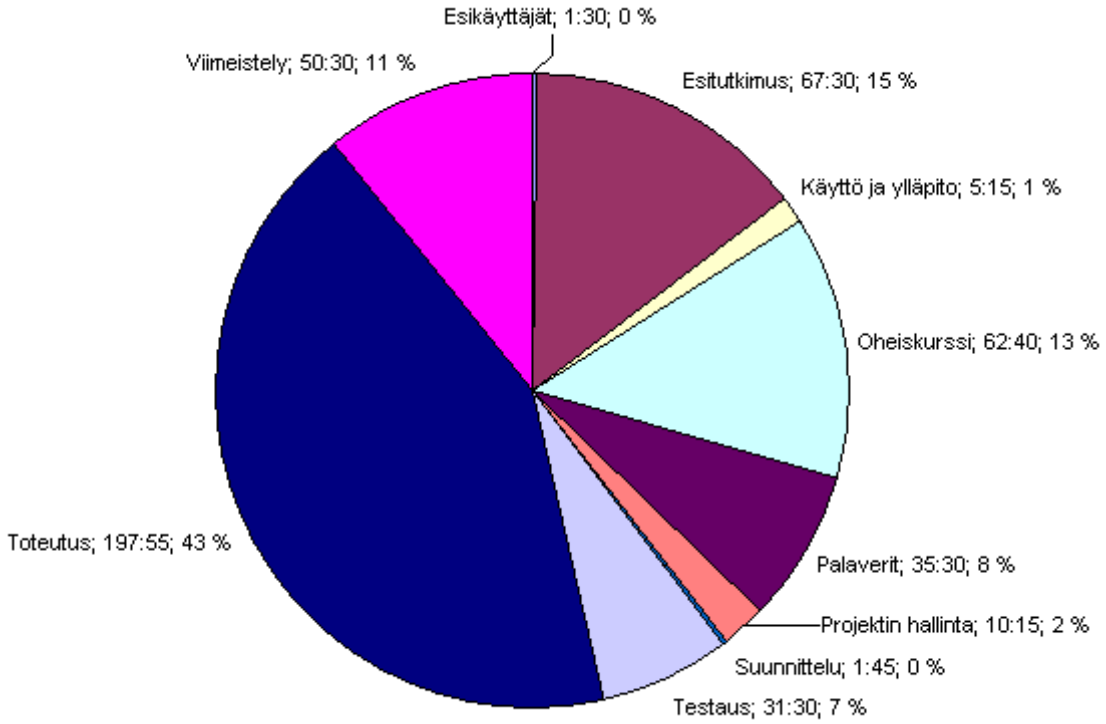
Kuva 10.2: Juhon ajankäyttö tehtäväkokonaisuuksittain

10.3 Tero Hännisen ajankäyttö tehtäväkokonaisuuksittain

Teron ajasta kolme neljännestä kului esitutkimukseen, sovellusta kehittämiseen ja viimeistelyyn. Verso-projektin toteutuspainotteisuus näkyy myös Terolla suurena toteutukseen käytettynä aikana (196 tuntia).

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

Tero laati palaveridokumenttien lisäksi lisäksi järjestelmätestaussuunnitelmaan liittyvät dokumentit [23], [24] ja [25], käytettävyyspäivän muistion [12], alustavertailun [16] sekä kokosi projektin kansion.



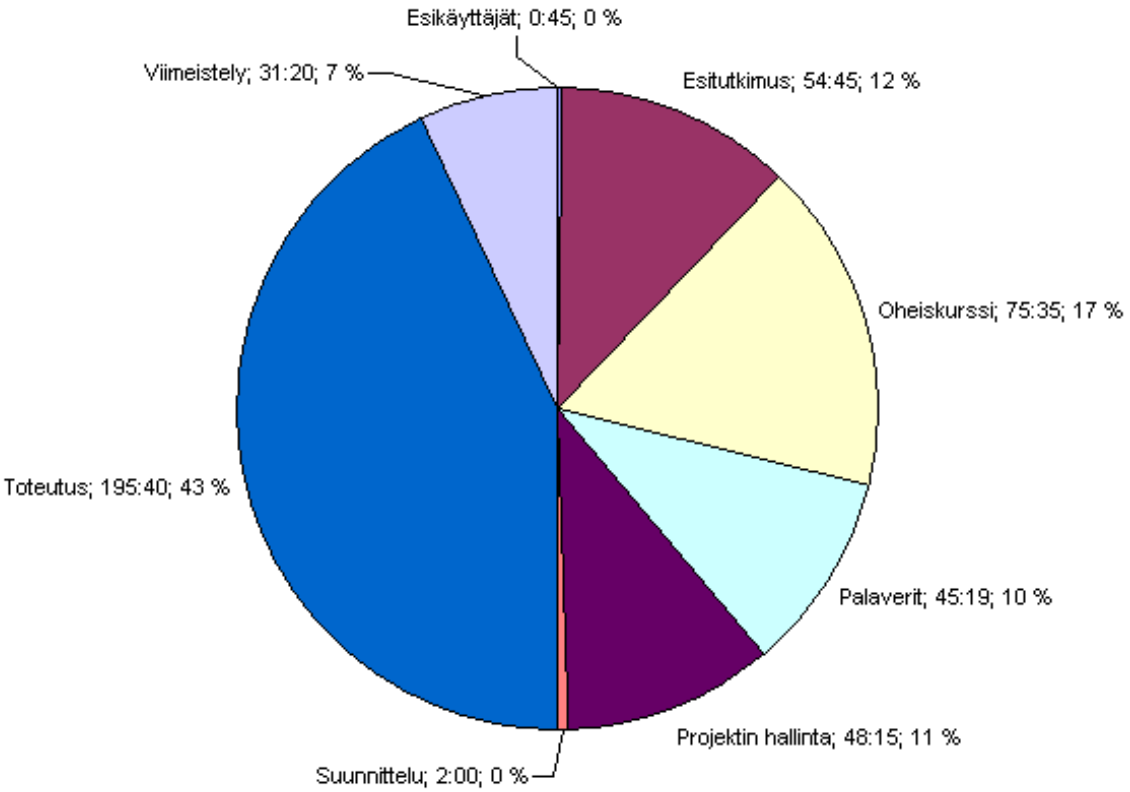
Kuva 10.3: Teron ajankäyttö tehtäväkokonaisuuksittain

10.4 Marko Peltolan ajankäyttö tehtäväkokonaisuuksittain

Markon ajasta kului Teron tavoin kolme neljännestä esitutkimukseen, sovelluksen kehittämiseen ja viimeistelyyn. Markokin käytti huomattavan suuren tuntimäärän sovelluksen kehittämiseen (195 tuntia).

Markon esitutkimukseen kulunut aika sisältää erityisesti projektissa käytetyn Ruby on Rails -sovelluskehyksen 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 palaveripöytäkirjojen ohella YouSource-sovellukselle asennusohjeen [17] ja 2. koodinkatselmoinnin muistion [3] sekä koosti Verso-projektin erillisen dokumentit sisältävän WWW-sivuston.



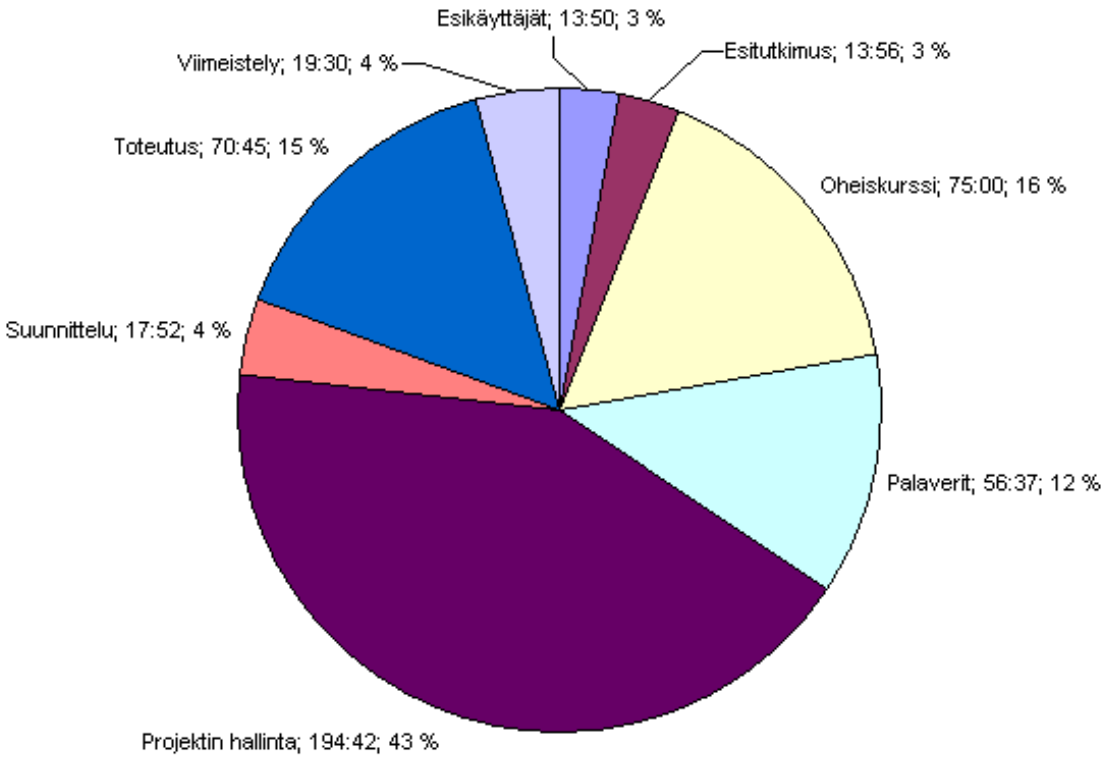
Kuva 10.4: Markon ajankäyttö tehtäväkokonaisuuksittain

10.5 Heikki Salon ajankäyttö tehtäväkokonaisuuksittain

Heikin Verso-projektin projektipäällikkyys näkyy korostuneena ajankäyttönä projektinhallinnassa (194 tuntia), josta valtaosa kului projektisuunnitelman ja -raportin laatimiseen.

Vaikka viivästyneen projektisuunnitelman laatiminen lopulta keskeytettiin, toteutukseen osuus (71 tuntia) on huomattava, sillä se on noin viidennes Heikin projektiin käyttämästä ajasta projektipäällikkönä. Ajankäytöstä on nähtävissä, että 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 [26] laatimiseen. Heikki laati palaveridokumenttien lisäksi projektiraportin [19], projektisuunnitelman [18], 1. koodikatselmoinnin muistion [2], tarvekuvauksen Korppi-ryhmien käyttämisestä [20] sekä KorppiLDAP:in käyttöä kuvaavan dokumentin [21].



Kuva 10.5: Heikin ajankäyttö tehtäväkokonaisuuksittain

11 Prosessimalli ja projektin aikataulu

Projekti oli määrä toteuttaa kevätlukukaudella 2010 siten, että sovellus on uusien ominaisuuksien toteuttamisen osalta valmis 23.4.2010 mennessä ja on kokonaisuudessaan toukokuun puoleen väliin mennessä. Uusien ominaisuuksien kohdalla aikataulu piti, mutta dokumentointi viivästyi noin puolella vuodella suunnitellusta. Aikataulua käydään läpi lisää kappaleesä kappaleessa 11.2.

Suunniteltuun verrattuna merkittävimmät erot liittyvät projektin raportoinnin viivästymiseen sekä projektipäällikön suunniteltua suurempaan teknisten tehtävien tekemiseen. Edellämainitut kaksi asiaa ovat myös kytköksissä keskenään. Projektisuunnitelman keskeyttäminen myöhäisessä vaiheessa ja jäädyttäminen hankaloittivat projektin läpivientiä sekä tekevät sen toteutumisen analysoimisesta vaikeaa tai paikoitellen mahdotonta.

11.1 Prosessimalli

Projekti oli määrä viedä läpi käyttäen ketterää ohjelmistokehitystä. Kesken jääneessä projektisuunnitelmassa [18] määriteltiin projektissa käytettävän projektille räätälöityä ketterää prosessimallia. Suunnitelmassa määriteltiin käytettävän kahden viikon mittaisia toteutusvaiheita, joiden kehitystehtävät sovitaan 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 järkevästi. Tehtävien kokojen arvioinnissa onnistuttiin pääosin hyvin, vaikka ensimmäisistä toteutusvaiheista jäi noin päivän tai kahden edestä valmistumattomia ominaisuuksia, josta lisää luvussa **??**. Valmistumattomat ominaisuudet hyväksyttiin palavereissa siirrettäväksi seuraavan vaiheen alussa.

Projekti vietiin läpi käyttäen **ketterää ohjelmistokehitystä**. Ominaisuus eli vaatimusmäärittelyssä Trac-tikettityyppi Feature hyväksytettiin 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 **määristä sopiminen vaiheiden aikataulun perusteella**. Käytännössä Verso-projektin sovelluksen kehittämiseen oli alusta asti varattu neljä kehitysvaihetta, joihon sisältyvät tehtävät valikoituivat vähitellen. Tämän esittäminen ei olisi ollut lainkaan mielekästä, 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 inkrementeittäin. Verso-projektiin sovitut vaiheet muistuttivat kuitenkin enemmän aikataulutettuja kehityspyrähdyksiä (engl. *sprint*).

Prosessimallin toteuma vastasi hyvin suunnitelmaa, joskin projektisuunnitelman [18] prosessimallikuvaus jäi suppeaksi. Laajemmin projektisuunnitelmassa kuvatut projektin Trac-tikettikäytänteet 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 kiinnittäessä. 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 työlistaa (engl. *product backlog*) tukevalla tikettijärjestelmällä, joka mahdollistaisi tehtävien suunnittelun korkeammalla tasolla. Tracin eduksi voi kuitenkin todeta sen toimineen yksittäisiä tehtäviä ajatellen hyvänä keskusteluympäristönä projektissa.

Tikettien kestoja arvioitiin ensimmäisen kolmen kehitysvaiheen ajan vain kokonaisuutena valittaessa kehitystehtäviä seuraaviksi kahdeksi viikoksi. Arviot pitivät paikkansa kohtuullisesti; 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 aloittavassa projektipalaverissa valittiin, mitkä ominaisuudet kehitetään ja mitkä rajataan jatkokehitykseen.

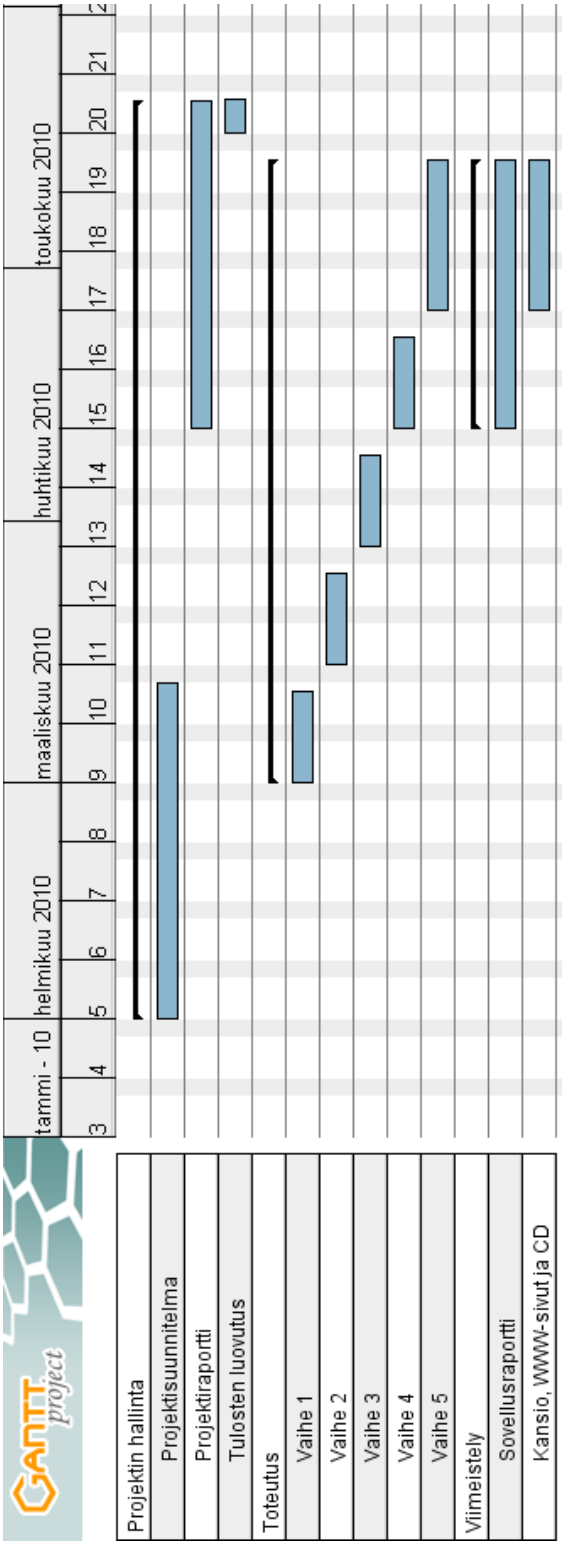
Projektiryhmän neljännelle vaiheelle arvoimat tikettien kestot pitivät keskimäärin paikkansa hyvin. Kun KorppiLDAP:in toteuttaminen ja näkyvyyksien muokkaaminen vaativat arvivoitua enemmään tunteja, muut valitut tehtävät valmistuvat nopeammin.

11.2 Aikataulu

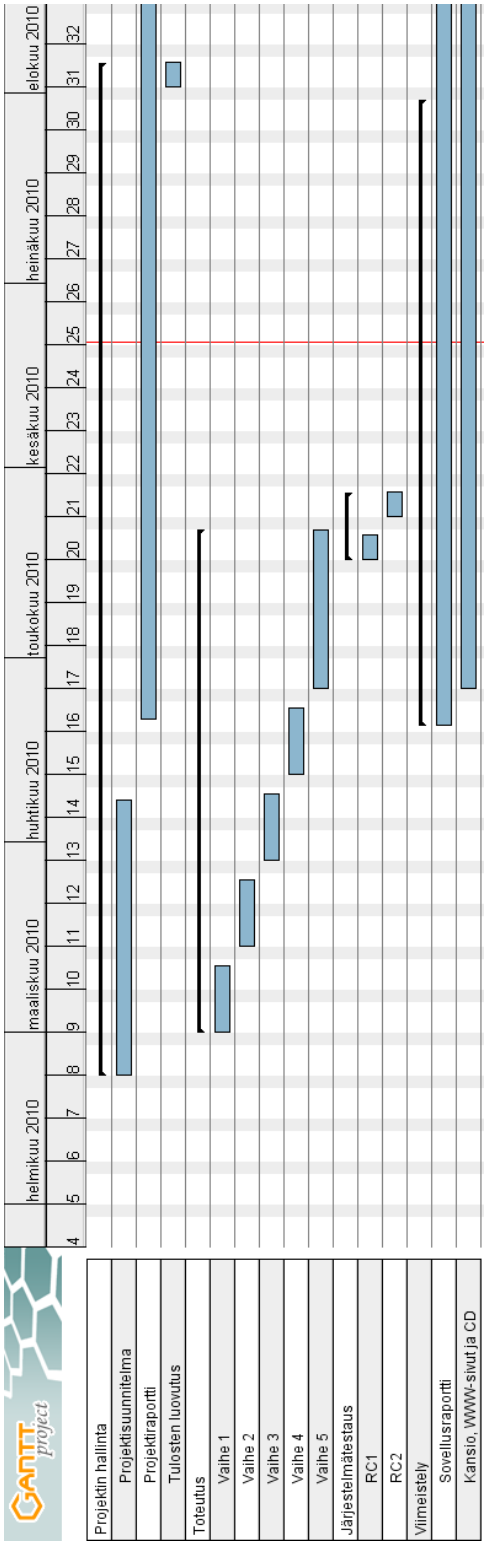
Projektin oli määrä olla kokonaisuudessaan valmis vuoden 2010 toukokuun puoleen väliin mennessä, mutta se myöhästyi siitä noin puolella vuodella. Sovelluksen osalta kehitystyö valmistui noin kahdella viikolla venynyttä viimeistelyvaihetta aikataulussa pääosin valitun prosessimallin perusteella.

Suunnitelman 11.1 ja toteuman 11.2 välillä **merkittävin ero liittyy viimeistelyyn ja raportointiin**, erityisesti raportointiin. Projektiraportin laatiminen jatkui kuvan ulkopuolella marraskuulle 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 vaiheittaisa myöhästymistä, sillä kunkin vaiheen myöhästyneet ominaisuudet hyväksyttiin osaksi edellistä seurannutta 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.

Tehtävät	**Juho**		**Marko**		**Tero**		**Heikki**		**Yhteensä**	
Vaihe	**S**	**T**	**S**	**T**	**S**	**T**	**S**	**T**	**S**	**T**
Projektin hallinta	**-**	**4**	**-**	**0**	**-**	**10**	**-**	**194**	**-**	**223**
muut tehtävät	-	4	-	0	-	10	-	10	-	24
projektisuunnitelma	-	0	-	0	-	0	-	93	-	93
projektiraportti	-	0	-	0	-	0	-	56	-	56
tiedotus	-	1	-	0	-	1	-	35	-	36
muut tehtävät	-	0	-	0	-	10	-	5	-	15
Esikäyttäjät	**10**	**7**	**10**	**1**	**10**	**2**	**32**	**14**	**62**	**24**
käytettävyystestaus	2	7	2	1	2	2	0	0	6	10
haastattelu	2	0	2	0	2	0	0	0	6	0
esikäyttötiedotus	0	0	0	0	0	0	8	1	8	1
esikäyttöviestintä	6	0	6	0	6	0	14	3	32	3
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**	**65**	**50**	**76**	**50**	**62**	**50**	**75**	**200**	**278**
esittelyt	15	30	15	27	15	21	15	20	60	98
katselmoinnit	5	3	5	3	5	3	5	6	20	15
koulutus	30	27	30	36	30	28	30	33	120	124
raportointi	-	5	-	10	-	10	-	16	-	41
Palaverit	**35**	**35**	**35**	**41**	**35**	**32**	**35**	**54**	**140**	**162**
raportointi	20	19	20	20	20	19	20	10	80	68
palaveri	15	16	15	21	15	13	15	44	60	94
Sovellusraportti	**50**	**76**	**5**	**0**	**5**	**0**	**5**	**0**	**65**	**76**
sovellusraportti	50	76	5	0	5	0	5	0	65	76
Suunnittelu	**-**	**6**	**-**	**2**	**-**	**2**	**-**	**16**	**-**	**26**
suunnittelu	-	6	-	2	-	2	-	16	-	26
Testaus	**-**	**0**	**.-**	**2**	**-**	**34**	**-**	**2**	**-**	**38**
järjestelmätestaus	-	0	.-	2	-	6	-	2	-	10
raportointi	-	0	.-	0	-	6	-	0	-	6
suunnittelut	-	0	.-	0	-	22	-	0	-	22
Toteutus	**50**	**263**	**-**	**213**	**-**	**196**	**-**	**65**	**-**	**737**
käyttöliittymä	-	93	-	8	-	19	-	2	-	122
metatiedostot	-	15	-	34	-	74	-	0	-	123
oikeudet ja näkyvyydet	-	5	-	99	-	21	-	0	-	125
päivitystavat	-	40	-	26	-	74	-	0	-	140
ohjeet ja neuvonta	-	0	-	20	-	0	-	0	-	20
ldap-autentikointi	-	23	-	0	-	0	-	17	-	40
tukitehtävät	-	11	-	26	-	8	-	46	-	91
Viimeistely	**-**	**4**	**-**	**20**	**-**	**44**	**-**	**13**	**-**	**81**
lähdekoodin siistiminen	-	4	-	20	-	44	-	13	-	81
Yhteensä	**195**	**518**	**100**	**410**	**100**	**450**	**117**	**460**	**467**	**1810**

Kuva 11.3: Jäsenten työmääräarvio

Taulukossa 11.3 on esitetty suunnitellut ja toteutuneet työmäärät sekä niiden jakau-

tuminen 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 mielekästä kuin osittain. Koko projektin tuntien hallintaan kiinnitettiin kunnolla huomiota vasta projektin loppupuolella.

Yleisesti projektiin käytettiin suunniteltua enemmän tunteja, Projektipäällikkö esitti projektin toisessa palaverissa [5] projektin tuntiarvioksi kultakin jäseneltä 250 tuntia, mitä pidettiin palaverissa heti matalana arviona. Projektin tuntien suunnittelu jäi keskeneräiseksi, mihin syinä olivat raportoinnissa käytetyn tavan ja itse projektin läpiviennin yhteensopimattomuus sekä projektipäällikön projektin alussa liian pieni panostus projektisuunnitelmaa kohtaan.

Projektin hallintaan käytettiin projektissa yhteensä 223 tuntia, joista merkittävin osa on kirjattu projektisuunnitelmalle ja -raportille. Projektisuunnitelmaan on kuitenkin merkitty myös projektisuunnitelman kieliasun korjaamiseen käytettyjä tunteja, jotka kirjauskäytänteiden päivittyessä on muissa dokumenteissa kohdennettu Oheiskurssi-kohdan alle.

Esikäyttäjät-kohdan toteutuneet tunnit jäivät paljon suunnitellusta, mikä johtui lähinnä odotettua vähäisemmästä esikäytöstä. Muutoin esikäyttöön suunnitellut tunnit toteutuivat odotetunkaltaisesti.

Kohtiin **Esitutkimus**, **Suunnittelu**, **Toteutus**, **Testaus** ja **Viimeistely** ei tunteja oltu suunniteltu laisinkaan. Suurelta osin tämän sivuuntuminen suunnitelmassa johtuu projektin tavasta käyttää määrämittaisia 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 edelliset muodostavat yli puolet toteutuneista tunneista.

Juhon laatima **sovellusraportti** suunniteltiin 50 laajuiseksi Juholle ja 5 tunnin laajuiseksi muille, mutta suunniteltu yhteinen sovelluksen toiminnan selvittäminen raportin laatimiseksi ei vaatinut erikseen merkittäviä rupeamia. Juhon käyttämä aika sovellusraporttiin oli noin puolet suunniteltua (50 tuntia) suurempi (76 tuntia).

11.3 Ryhmän ajankäyttö viikoittain

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

Viikottaisen ajankäytön kuvassa 11.4 näkyy selvästi neljä trendiä:

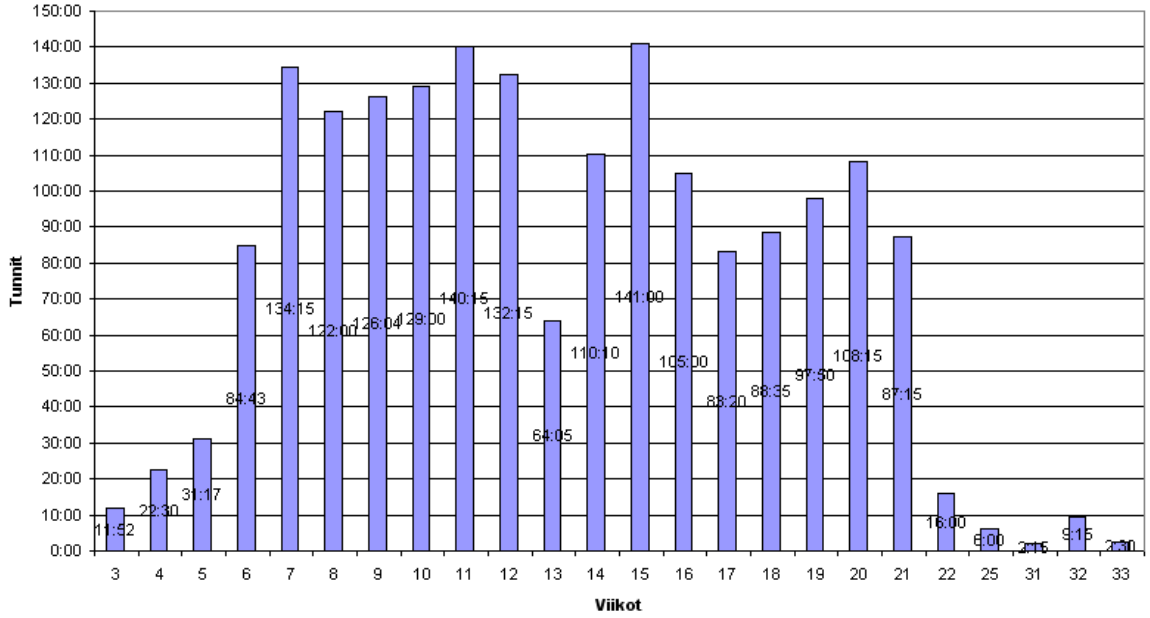
- projektin alku ennen tilaajaa ja toimeksiantoa (viikot 3-5)
- kehityksen alkupuoli (viikot 5-15)
- kehityksen loppupuoli (viikot 16-21) ja
- viimeistely (viikot 21 ja eteenpäin).

Projektin alussa kertyneet 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 ryhmäläiset olivat käyttäneet projektiin tunteja omaa mukavuustasoaan enemmän. Lisäksi sovelluksen nähtiin lisäksi voivan valmistua pienemmilläkin tuntimäärillä. Kehitystä olisi pitänyt kuitenkin karsia dokumentoinnille tilaa antaen, sillä ryhmän jäsenten kesätöiden alettua viikolla 22

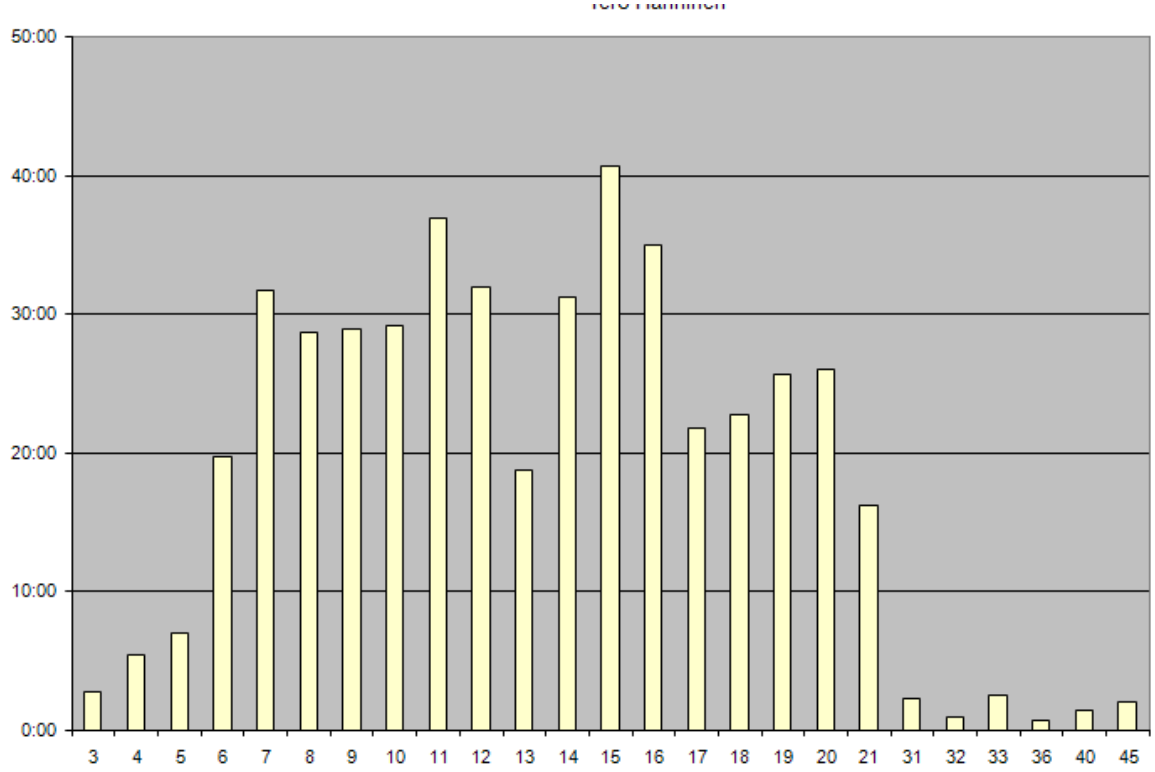
ensimmäisen viikolla 6 pidetyn palaverin jälkeen alkanut perehtyminen ja alustavaihtoehtojen kartoittaminen. Viikoilla 13 ja 14 näkyy Juhon sairastumisen ja pääsiäispäivien aiheuttama vaje tunneissa. Myös ryhmän kesätöiden alkaminen näkyy selvästi romahduksena tunneissa viikolla 22. Kuvassa 11.4 ei edes näytetä kaikkia viikkoja vuoden 2010 ajalta, todellisuudessa ei-aktiivisia viikkoja on ollut useita.

11.4 Tero Hännisen ajankäyttö viikoittain

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



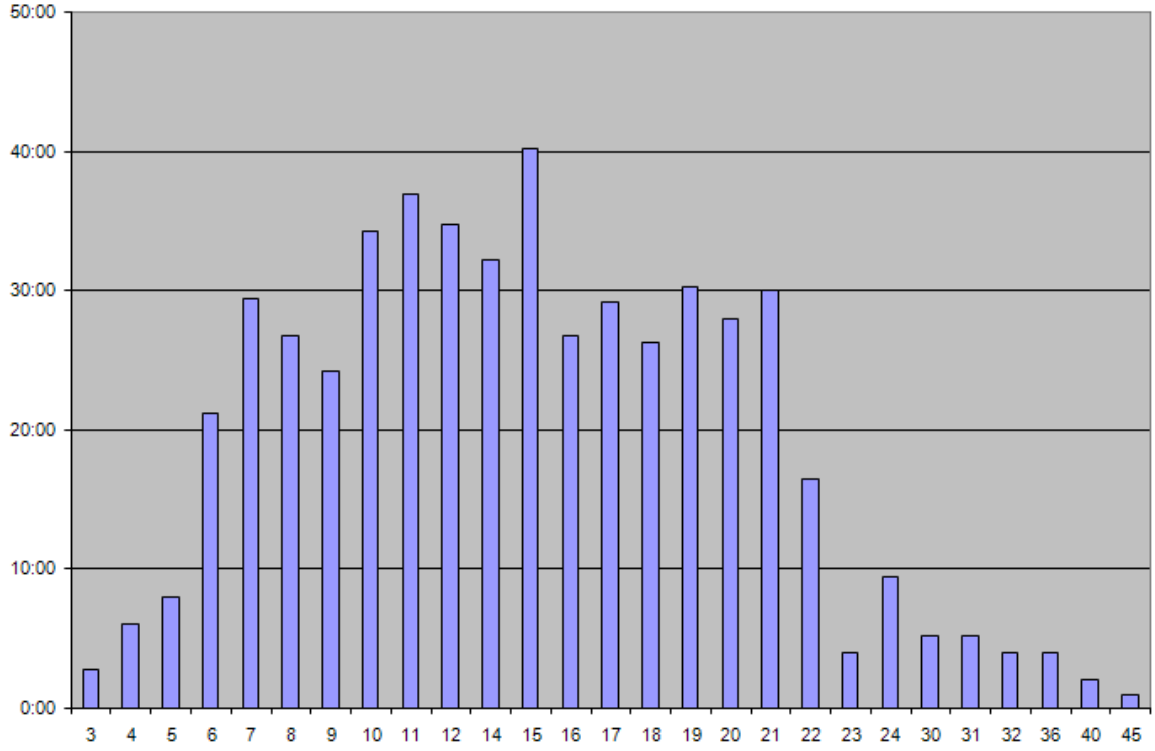
Kuva 11.4: Ryhmän ajankäyttö viikoittain



Kuva 11.5: Teron ajankäyttö viikoittain.

11.5 Juho Niemisen ajankäyttö viikoittain

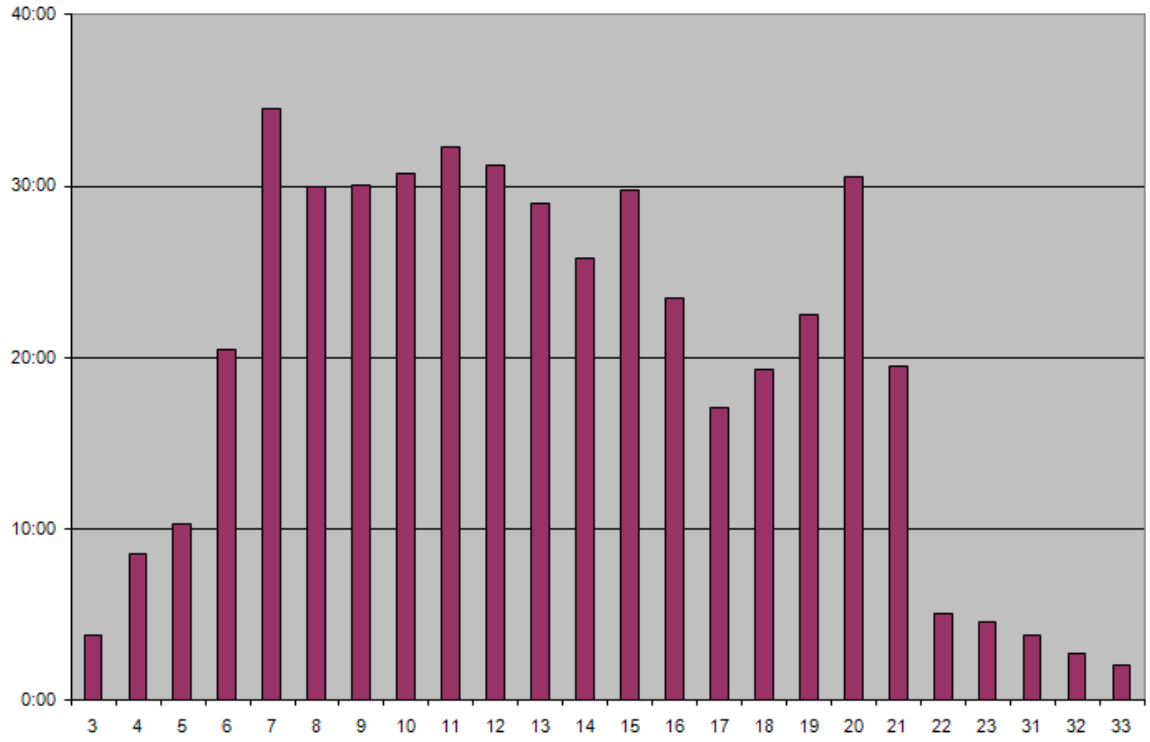
Juhon ajankäyttö noudattelee myös ryhmän ajankäyttöä. Kuvasta 11.6 puuttuu Juhon sairastumisen takia kokonaan pääsiäisviikko 13. Juhon kesätyö alkoi viikolla 21, mikä näkyy graafissa tuntimäärän vähenemisenä. Juhon vastuisiin projektin loppupuolella kuului sovellusraportin laatiminen, mikä näkyy Teroa ja Markoa suurempana ajankäyttönä loppukesällä ja syksyllä.



Kuva 11.6: Juhon ajankäyttö viikoittain.

11.6 Marko Peltolan ajankäyttö viikoittain

Myös Markon ajankäyttö noudattelee ryhmän yleistä ajankäyttöä. Marko aloitti Teron kanssa projektissa toteutetun sovelluksen jatkokehityken viikolla 21, mikä näkyy tuntimäärien vähenemisenä. Markon vastuisiin kuului projektin loppupuolella Verso-projektin verkkosivuston kokoaminen palvelimelle `http://sovellusprojektit.it.jyu.f` ja sovelluksen asennusohjeen viimeistely.

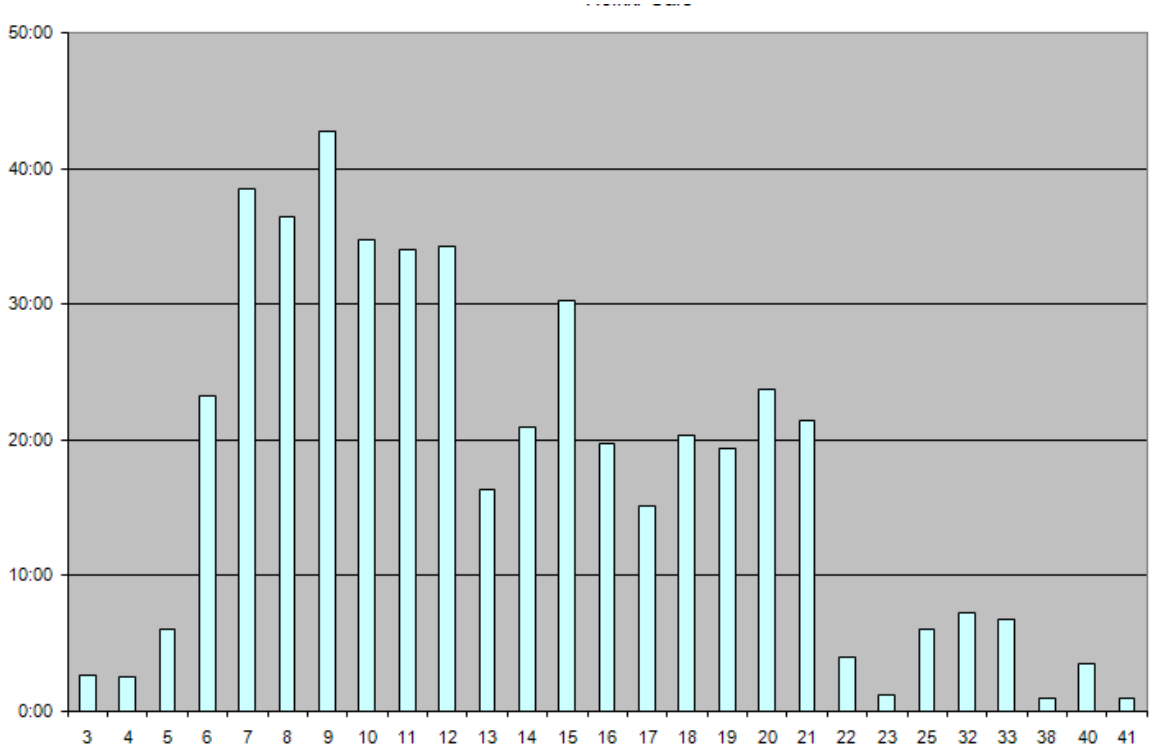


Kuva 11.7: Markon ajankäyttö viikoittain.

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 viikoittaisien tuntimäärien vähenemisenä. Heikillä oli projektin projektin loppuunsaattamisen organisointitehtävien ohella laadittavana projektiraportti, mikä näkyy käytettyinä tunteina kesällä ja syksyllä.



Kuva 11.8: Heikin ajankäyttö viikoittain.

12 Riskit ja niiden seuranta

Luvussa kuvataan projektin ennakoitujen ja ennakoimattomien riskien toteutumista sekä vaikutuksia tuloksiin tai projektiin. Keskeneräinen projektisuunnitelma [18] 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 toteuttaessa. Uusien työkaluvaatimuksien kanssa projektiryhmä oli riippuvainen 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-laajennosta 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. Projektipäällikkönä tässä kohtaa on todettava, että suurempaa motivaatiota tekemistä 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 sen nähtiin lupaavimmaksi tavoitteiden saavuttamisen kannalta projektiryhmän suorittamassa alustavertailussa [16]. Projektin kuluessa oli silti mahdollista, että Gitoriouksen käyttö alustana estyy 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 -sovelluskehykseen. Riski olisi toteutuessaan olennaisesti hankaloittanut sovellukselle asetettujen tavoitteiden saavuttamista. Riskin vaikutus arvioitiin pieneksi, ja sen vaikutus projektiin jäi myös pieneksi. Ryhmä korjasi Gitoriouksesta yhden YouSourcea kehittäessä esiintulleen tietoturvapuutteen, joka korjautui samalla myös Gitorious-kantasovellukseen.

Riskiä ei varsinaisesti voitu kontrolloida, mutta projektiryhmä seurasi koko projektin ajan kaikkia Gitoriouksen 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 keväälle viikonloppua pidempiä lomamatkoja keväälle esimerkiksi sairastumisista johtuvia poissaoloja ollut aihetta odottaa ennalta. Riskiä

hallittiin projektin alusta loppuun kestäneellä hyvällä kommunikoinnilla jäsenten välillä. 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 toimisi tuntemattomasta syystä väärin.

Projektissa toteutettava sovellus sisältää paljon erilaisia komponentteja, joiden yhteistoiminnasta sovellus on riippuvainen. Jos sovelluksen jokin osa vikaantuu ja alkaa toimia väärin, tilanteen selvittäminen on mahdollisesti haastavaa ja aikaavievää, mikä viivästyttää kehitystyötä.

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

Riski toteutui vain pienenä, koska ongelmatilanteiden ratkaisussa ei yleensä mennyt pitkään. Projektisuunnitelmassa [18] kuvattiin riskin hallitsemiseksi harjoitteet erilaisten vikatilanteiden etukäteen harjoituttamiseksi. Suunnitellusta kuitenkin poikettiin, sillä vastaantulleet ja mahdollisesti yleistyvät vikatilanteet korjattiin yleensä koko projektiryhmällä ja tietämys korjaustoimenpiteistä levisi saman tien. Tarvetta erilliselle harjoittamiselle ei 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 etu-

käteen lainkaan, eikä se toteutunut, sillä esikäyttäjien aktiivisuus oli vähäinen, kuten luvussa 7.5 kuvataan.

13 Projektiryhmäläisten kokemuksia

Luvussa kuvataan projekilaisten kokemuksia Verso-projektista. Projekti onnistui laajuuteensa nähden hienosti. Projekti sisälsi ryhmän kannalta monia uusia asioita: uudet tekniset työkalut, tilaajan kanssa toimiminen, projektikäytänteet, ATK-tuen ja sovelluspalveluiden kanssa toimiminen ja niin edelleen.

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 ajatukseen lähdekoodien julkistamisjärjestelmästä ja siitä, mitä järjestelmällä oikeastaan halutaan julkistaa.

Kolmas muutostoive liittyy dokumentteihin. Esimerkiksi projektisuunnitelma olisi ehdottomasti ensin pitänyt luonnostella erillisiin pieniin, 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 ja olenkin todella tyytyväinen, että ne tuli opittua 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 jonkinnäköinen prototyyppi, mutta tuloksena oli toimiva ja käytettävä sovellus. Tähän toki auttoi se, että sovelluksen pohjaksi löytyi paljon 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 sovellusprojektikurssini 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 käytettävyystestaukset 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 pänttääminen 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 ohjelmistokehyksestä, 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. En pidä kirjoittamisesta, enkä esiintymisestä, mutta sain näilläkin osa-alueilla arvokasta kokemusta, josta on vielä tulevaisuudessa varmasti hyötyä. Minulle henkilökohtaisesti tärkeimpänä sovellusprojektissa oli kui-

tenkin, 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 härkäpäisesti 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 dokumenttikeskeisyyteen: sillä jokaisen ilmaistun ajatuksen on löydyttävä asiakirjamuodossa, oikoluettuna ja kauniiksi ladottuna. Dokumenttikeskeisyys näkyy myös siten, että sovellusprojektikurssilla keskitytään ohjaaamaan asiakirjojen, ei ajatusten kehitystä. Täytyy tunnustaa, että ennalta kuulemani perusteella suhteeni dokumentteihin oli kriisiytynyt jo projektin alkaessa Jukka-Pekka Santasen ohjaamana. Sovellusprojektikurssin sisällössä on aihetta Verso-projektia laajemmallekin keskustelulle.

Sovellusprojektikurssista jää ehdottomasti mieluisimpina mieleen hyvä ja toimeentuleva ryhmä sekä mielenkiintoinen aihe. Voisi sanoa, että kaikki ryhmään, tekniikkaan ja sovellukseen, siis kaikenkaikkiaan tekemiseen liittyvä sovellusprojektikurssilla, toimi hienosti, 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 välillä. Ryhmä sai myös kokemusta useista sovelluskehityksessä tarvittavista työkaluista sekä 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 sovellus- ja projektiraporttien valmistuttua 1.1.2011. Projekti myöhästyi suunnitellusta noin puoli vuotta myöhässä, sillä projektin viimeistelylle ja raporteille ei oltu varattu riittävästi aikaa.

15 Lähteet

[1] Petri Heinonen, Ajankäytönseurantasovellus, saatavissa Excel-muodossa <URL: `http://appro.mit.jyu.fi/tools/ajankaytto/ajankaytonseuranta.xls`>, Jyväskylän yliopisto, informaatioteknologian tiedekunta, viitattu 11.11.2010.

[2] Heikki Salo, "1. koodinkatselmoinnin muistio", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/katselmoinnit/katselmointi1.txt>, Verso-projekti.

[3] Marko Peltola, "2. koodinkatselmoinnin muistio", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/katselmoinnit/katselmointi2.txt>, Verso-projekti.

[4] Juho Nieminen, "1. palaverin pöytäkirja", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/palaverit/poytakirja_1_palaveri.txt>, Verso-projekti.

[5] Juho Nieminen, "2. palaverin pöytäkirja", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/palaverit/poytakirja_2_palaveri.txt>, Verso-projekti.

[6] Marko Peltola, "4. palaverin pöytäkirja", saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/palaverit/poytakirja_4_palaveri.txt>, Verso-projekti.

[7] Ville Tirronen, "Alustavaa lukemistoa projektiin liittyen".

[8] "About Gitorious", saatavissa HTML-muodossa <URL: http://gitorious.org/about>.

[9] Wikipedia, "Gitorious", saatavissa HTML-muodossa
<URL: http://en.wikipedia.org/wiki/Gitorious>.

[10] Gitorious, "Gitorious", saatavissa XHTML-muodossa
<URL: http://groups.google.com/group/gitorious?pli=1>.

[11] Gitorious, "Gitorious Coding Style Guide", saatavissa HTML-muodossa
<URL: http://gitorious.org/gitorious/pages/CodingStyleGuide>.

[12] Tero Hänninen, “Käytettävyyspäivän muistio”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/kaytettavyys_dokumentit/kaytettavyyspaiva.txt>, Verso-projekti.

[13] Juho Nieminen, “1. käytettävyystestauksen muistio”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/kaytettavyys_dokumentit/kaytettavyystestaus_1.txt>, Verso-projekti.

[14] Juho Nieminen, “1. käytettävyystestauksen muistio”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/kaytettavyys_dokumentit/kaytettavyystestaus_2.txt>, Verso-projekti.

[15] “Ruby Style Guide”, saatavissa HTML-muodossa
<URL: http://github.com/chneukirchen
/styleguide/raw/master/RUBY-STYLE>.

[16] Tero Hänninen, “Vertailu alustoista”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/vertailu_alustoista.txt>, Verso-projekti.

[17] Marko Peltola, “YouSource RHEL installation guide”, saatavissa HTML-muodossa
<URL: http://yousource.it.jyu.fi/yousource
/pages/Installation_Guide>.

[18] Heikki Salo, “Projektisuunnitelma”, saatavissa PDF-muodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/projektisuunnitelma
/Verso_projektisuunnitelma_0.6.0.pdf>, Verso-projekti.

[19] Heikki Salo, “Projektiraportti”, saatavissa PDF-muodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/projektiraportti
/Verso_projektiraportti_1.0.pdf>, Verso-projekti.

[20] Heikki Salo, “Tarvekuvaus Korppi-ryhmien käyttämisestä”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/tarvekuvaus_korppi-ryhmien_kayttamisesta.txt>, Verso-projekti.

[21] Heikki Salo, “YouSource-sovellukseen kirjautuminen KorppiLDAPilla”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/yousource-sovellukseen_kirjautuminen_korppildapilla.txt>, Verso-projekti.

[22] Juho Nieminen, “Sovellusraportti”, saatavissa PDF-muodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/application/application_report/application_report_1.0.pdf>, Verso-projekti.

[23] Tero Hänninen, “System test plan”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/application/system_tests/system_test_plan_21st_May.txt>, Verso-projekti.

[24] Tero Hänninen, “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>, Verso-projekti.

[25] Tero Hänninen, “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>, Verso-projekti.

[26] Heikki Salo, “Suunnitelma esikäyttäjien hyödyntämisestä”, saatavissa raakatekstimuodossa
<URL: http://sovellusprojektit.it.jyu.fi/verso/src
/docs/project/suunnitelma_esikayttajien_hyodyntamisesta.txt>, Verso-projekti.

[27] Antti-Juhani Kaijanaho, “Teknisen ohjaajan huomioita koodista”, saatavissa HTML-muodossa

<URL: https://korppi.jyu.fi/kotka/servlet/list-archive
/verso/0771.html>, Verso-projekti.

[28] Gitorious, "Usability suggestions to dashboard vs. profile, activity feed and projects", saatavissa HTML-muodossa
<URL: http://groups.google.com/group/gitorious
/browse_thread/thread/a6a888ff10b770ef/>, Verso-projekti.

[29] Verso-projekti, "Verso-projektin tehtävähallinta", saatavissa HTML-muodossa
<URL: https://trac.cc.jyu.fi/projects/verso>.