Lappeenrannan teknillinen yliopisto
Teknistaloudellinen tiedekunta
Tietotekniikan koulutusohjelma

Diplomityö

Joonas Äijälä

SHAREPOINT-SOVELLUSTEN KÄYTETTÄVYYS JA SEN KEHITTÄMINEN

Työn tarkastajat: Professori Kari Smolander
Diplomi-insinööri Antti Roine

Työn ohjaaja: Diplomi-insinööri Antti Roine

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto
Teknistaloudellinen tiedekunta
Tietotekniikan koulutusohjelma

Joonas Äijälä

SharePoint-sovellusten käytettävyys ja sen kehittäminen

Diplomityö

2012

91 sivua, 9 kuvaa, 3 taulukkoa, 3 liitettä

Työn tarkastajat: Professori Kari Smolander
Diplomi-insinööri Antti Roine

Hakusanat: käytettävyys, SharePoint, monipuoliset Internet-sovellukset
Keywords: usability, SharePoint, rich Internet applications

Tässä työssä tutkitaan SharePoint-sovellusten käytettävyyden kehittämistä. Käytettävyyden kehittämiselle etsitään keinoja kirjallisuudessa esitetyistä teoreettisista keinoista. Lisäksi selvitetään monipuolisten Internet-sovellusten käytön vaikutuksia SharePoint-sovellusten käytettävyyteen. Tässä työssä toteutetaan SharePoint-ohjelmisto, joka sisältää kaksi erilaista käyttöliittymää. Näitä käyttöliittymiä vertaillaan keskenään käytettävyyden näkökulmasta ja havaitaan, että käyttäjäpalautteen ja monipuolisten Internet-sovellusten ansiosta saadaan toteutettua laadukkaampaa käytettävyyttä. Työssä tehdään kyselytutkimus SharePointia käyttäville yrityksille, jossa selvitetään SharePoint-sovellusten käytettävyyden merkitystä ja laatua. Kyselyyn saadaan vain vähän vastauksia, jonka takia sen pohjalta ei pystytä tekemään yleistäviä johtopäätöksiä.

ABSTRACT

Lappeenranta University of Technology
Faculty of Technology Management
Degree Program in Information Technology

Joonas Äijälä

The usability of SharePoint-applications and how to improve it

Master's Thesis

91 pages, 9 figures, 3 tables, 3 appendices

Examiners : Professor Kari Smolander
Master of sciences Antti Roine

Keywords: usability, SharePoint, rich Internet applications

In this thesis we perform research on how to improve the usability of SharePoint-applications. Means of usability improvement are found from theoretical guides in literary sources. Additionally the effects of rich Internet applications on the usability of SharePoint-applications is inspected. In this thesis a SharePoint software containing two separate user interfaces is implemented. These user interfaces are compared with each other from the point of view of usability, and it is found that due to both user feedback and rich Internet applications it is possible to achieve better usability. In the thesis a survey is sent to companies using SharePoint, in which the importance and quality of usability in SharePoint-applications is queried on. The survey receives only a few answers, because of which no generalizations could be made from the answers.

ALKUSANAT

Tämä työ on tehty Onsight Helsinki Oy:ssä. Kiitän työn ohjaajaa mahdollisuudesta tämän työn tekoon, sekä kaikkia tässä työssä tehtyyn kyselytutkimukseen vastanneita henkilöitä lähetetyistä vastauksista.

SISÄLLYSLUETTELO

1 JOHDANTO 4

1.1 TAUSTA 4

1.2 TAVOITTEET JA RAJAUKSET 5

1.3 TYÖN RAKENNE 6

2 TYÖSSÄ KÄSITELTÄVIEN AIHEIDEN TEORIA 8

2.1 KÄYTETTÄVYYS 8

2.1.1 Käytettävyyden arviointi 10
2.1.2 Käytettävyyden kehittäminen 13

2.2 SISÄLLÖNHALLINTAJÄRJESTELMÄT JA YHTEISTYÖALUSTAT 16

2.3 MONIPUOLISET INTERNET-SOVELLUKSET 22

2.3.1 Monipuolisten Internet-sovellusten teknologiat ja kehittäminen 23
2.3.2 Monipuoliset Internet-sovellukset ja käytettävyys 26

3 TYÖSSÄ KÄYTETYT TEKNOLOGIAT 30

3.1 SHAREPOINT 30

3.1.1 SharePoint sisällönhallintajärjestelmänä 31
3.1.2 SharePoint yhteistyöalustana 34
3.1.3 SharePoint ohjelmistoalustana 36

3.2 SILVERLIGHT 41

4 TYÖSSÄ KÄYTETYT MENETELMÄT JA TYÖTAVAT 44

4.1 ASIAKASKYSELY 44

4.2 TYÖJONO-OHJELMAN SUUNNITTELU JA TOTEUTUS 44

5 ASIAKASKYSELY SHAREPOINTISTA JA KÄYTETTÄVYYDESTÄ 45

6 CASE: TYÖJONO 48

6.1 TYÖJONON SUUNNITTELU 48

6.2 TYÖJONON TOTEUTUS 52

6.2.1 Työjonon ASP.NET-käyttöliittymä 53
6.2.2 Työjonon Silverlight-käyttöliittymä 54

6.3 TYÖJONON KÄYTETTÄVYYS 57

7 KESKUSTELU 65

8 YHTEENVETO..... 68

LÄHTEET..... 69

LIITTEET

SYMBOLI- JA LYHENNELUETTELO

AJAX	Asynchronous JavaScript and XML
BCS	Business Connectivity Services
CSS	Cascading Style Sheet
EWS	Exchange Web Services
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IIS	Internet Information Services
ISO	International Organization for Standardization
REST	Representational State Transfer
RIA	Rich Internet Application
TOTE	Test, Operate, Test, Exit
WCF	Windows Communication Framework
XAML	Extensible Application Markup Language
XML	Extensible Markup Language

1 JOHDANTO

1.1 Tausta

Käytettävyys on tietokoneohjelmistoihin liitetty ominaisuus, joka liittyy muun muassa ohjelmistojen helppokäyttöisyyteen ja käytön tehokkuuteen [1]. Käytettävyydeltään laadukkaat ohjelmistot tarjoavat käyttäjilleen lukuisia erilaisia etuja verrattuna ohjelmistoihin, joiden käytettävyys on heikkolaatuista. Yrityksissä ja organisaatioissa nämä edut puolestaan ilmenevät rahallisina säästöinä. Tästä johtuen ohjelmistoyritysten asiakasorganisaatioiden on liiketoiminnallisesti kannattavampaa hankkia käytettävyydeltään laadukkaita ohjelmistoja. [2] Täten ohjelmistoja ostavien organisaatioiden voidaan olettaa hankinnoissaan suosivan tuotteita, jotka tarjoavat hyviä käytettävyysominaisuuksia.

Koska laadukas käytettävyys voi tehdä ohjelmistosta haluttavamman asiakkaiden näkökulmasta, voi ohjelmistoyritysten olla kannattavaa panostaa käytettävyyteen myös niiden oman liiketoiminnan näkökulmasta. Käytettävyydeltään hyviä ohjelmistoja voi olla helpompi myydä tai niistä voidaan saada parempaa hintaa, kuin huonosti käytettävistä ohjelmistoista. Samalla asiakkaat, jotka ovat tyytyväisiä hankkimiinsa tuotteisiin, saattavat jatkossakin ostaa ohjelmistotototeutuksia samalta toimittajalta. Muun muassa näistä syistä ohjelmistoyrityksissä voisi olla kannattavaa panostaa toteutettavien ohjelmistojen käytettävyyden kehittämiseen.

Microsoftin kehittämä SharePoint-ohjelmisto on yleistynyt yritysten käytössä, ja sen käyttöönottojen määrä on lisääntynyt merkittävästi viime aikoina [3]. Samalla kun SharePoint yleistyy yritysten käytössä, voidaan olettaa myös yritysten tarpeiden SharePoint-ohjelmistoprojekteille lisääntyvän. Samalla SharePointin merkitys ohjelmistoalustana ohjelmistotoimittajien keskuudessa suurenee, ja tarve entistä laadukkaampien SharePoint-toteutusten toimittamiselle lisääntyy. Tämän seurauksena ohjelmistotoimittajien voisi olla kannattavaa kehittää toteuttamiensa SharePoint-sovellusten käytettävyyttä. Selvitettäväksi jääkin, miten SharePoint-sovellusta käytettävyyttä voidaan kehittää laadukkaampien ohjelmistotuotteiden valmistamiseksi?

1.2 Tavoitteet ja rajaukset

Tämän työn tavoitteena on selvittää SharePoint-sovellusten käytettävyyden tilaa, sekä tutkia miten SharePoint-sovellusten käytettävyyttä voidaan kehittää. Yleisesti ottaen, tavoitteena on vastata seuraaviin tutkimuskysymyksiin:

- Miten laadukkaaksi SharePointin käyttäjät kokevat SharePointin käytettävyyden, ja kuinka tärkeäksi käytettävyys koetaan?
- Miten SharePoint-sovellusten käytettävyyttä voidaan parantaa, ja voidaanko SharePoint-sovelluksissa saavuttaa parempaa käytettävyyttä hyödyntämällä monipuolisten Internet-sovellusten teknologioita?

Selvitettäessä SharePointin käyttäjien kokemuksia käytettävyydestä tavoitteena on saada sekä yleistä, helposti tilastoitavaa tietoa, että tietoa sellaisista SharePointiin liittyvistä ongelmista, jotka käyttäjät itse kokevat merkittäviksi. Käytettävyyden merkitystä ja laatua on tarkoitus selvittää sellaisenaan, eikä minkään muun ohjelmistoihin liitettävän ominaisuuden vastakohtana. Käyttäjien kokemuksia selvitettäessä tarkoituksena ei myöskään ole tutkia käytettävyyttä yksityiskohtaisesti. Sen sijaan käytettävyyttä ja siihen liittyviä kokemuksia on tavoitteena käsitellä yleisellä tasolla.

Käytettävyyden kehittämistä on tavoitteena selvittää sekä teorian että käytännön näkökulmista. Teoreettisessa tarkastelussa pyritään kirjallisuuden avulla löytämään erilaisiin lähtökohtiin perustuvia tapoja käytettävyyden parantamiseen sekä käytettävyyden laadun arvioimiseen. Näitä kirjallisuudessa esitettyjä keinoja on tarkoitus käsitellä sekä teoreettisina ideoina, irrallaan toteutuksissa käytettävistä teknologioista, että osana monipuolisten Internet-sovellusten mahdollistamia toteutuksia. Käytäntöön perustuvassa tutkimuksessa tavoitteena on löytää tapoja, joilla käytettävyyttä voidaan parantaa, ja joilla tarpeita käytettävyyden parantamiselle voidaan löytää. Lisäksi pyritään selvittämään, miten SharePoint-sovellusten käytettävyyttä voidaan käytännössä parantaa hyödyntämällä monipuolisten Internet-sovellusten teknologioita, ja miten parempi käytettävyys voi ilmetä käytännössä. Tavoitteena ei ole löytää yksityiskohtaisia ohjeita, joita seuraamalla ohjelmistotototeutuksissa olisi mahdollista saavuttaa parempaa käytettävyyttä. Sen sijaan tarkoituksena on löytää yleisen tason sovellettavia keinoja laadukkaamman käytettävyyden

toteuttamiseksi SharePoint-ohjelmistoprojekteissa.

Tässä työssä tehtävä käytettävyyden tarkastelu, joka ei ole irrallaan ohjelmistototeutuksissa käytettävistä teknologioista, tehdään vain SharePoint-ohjelmistokehitykseen sovellettavissa olevissa asiayhteyksissä. Toisin sanoen, käytettävyyttä tarkastellaan vain teoreettisesti, osana monipuolisten Internet-sovellusten kehittämistä sekä osana SharePointia. Käytettävyyttä ei siis tulla käsittelemään esimerkiksi SharePointia muistuttavien tuotteiden osalta, vaikka tällaisista tuotteista saatavat tulokset saattaisivatkin olla sovellettavissa myös SharePointiin. SharePointista saatavia tuloksia ei myöskään tulla yleistämään muihin tuotteisiin.

Käytettävyyttä käsitellään tässä työssä irrallaan muista ohjelmistoihin liitettävistä ominaisuuksista, eikä sitä tulla laittamaan vastakkain näiden ominaisuuksien kanssa. Käytettävyyden tarkastelussa ei tulla ottamaan kantaa käytettävyyden kehittämisen mielekkyyteen erilaisissa ohjelmistoprojekteissa. Sen sijaan laadukkaasta käytettävyydestä saatavia hyötyjä tullaan selvittämään lyhyesti. Tässä työssä ei siis ole tavoitteena vastata kysymykseen milloin käytettävyyttä kannattaa kehittää.

1.3 Työn rakenne

Tämän työn luvussa 2 esitellään työhön liittyvät teoreettiset aiheet kirjallisuuden näkökulmasta. Tässä luvussa käsiteltäviä asioita ovat käytettävyys sekä sen arvioiminen ja kehittäminen, sisällönhallintajärjestelmät, yhteistyöalustat sekä monipuoliset Internet-sovellukset, niiden kehittäminen ja niiden vaikutus sovellusten käytettävyyteen. Luvussa 3 esitellään tähän työhön liittyvässä käytännön ohjelmistototeutuksessa käytetyt teknologiat ja tuotteet. Näitä tuotteita käsitellään sellaisenaan, sekä osana niiden tuotteiden joukkoja, jotka esitellään luvussa 2.

Luvussa 4 esitellään työn käytännön osioissa käytetyt työtavat ja menetelmät. Luvussa 5 käsitellään tätä työtä varten toteutettu kyselytutkimusta sekä käydään läpi kyseisen tutkimuksen tuloksia. Tässä luvussa kyselyn tuloksista tehdään myös alustavia johtopäätöksiä. Luvussa 6 esitellään tähän työhön liittyvä käytännön ohjelmistototeutus, siihen liittyvät suunnittelu- ja toteutusvaiheet ja toteutuksen lopputuloksina saadut

ohjelmiston käyttöliittymät. Ohjelmiston käyttöliittymiä vertaillaan keskenään käytettävyyden näkökulmasta, ja niiden käytettävyyden laadusta ja siihen vaikuttaneista tekijöistä tehdään johtopäätöksiä.

Luvussa 7 kerrataan työssä saatuja yksittäisiä tuloksia ja niiden pohjalta tehdään johtopäätöksiä. Työssä tehtyjen havaintojen valossa ehdotetaan myös mahdollisia tarpeita jatkotutkimukselle. Luvussa 8 esitetään yhteenveto tämän työn vaiheista sekä tässä työssä saaduista lopputuloksista.

2 TYÖSSÄ KÄSITELTÄVIEN AIHEIDEN TEORIA

2.1 Käytettävyys

Käytettävyys on käsite, jota voidaan tulkita monella tapaa tai joka voidaan helposti sekoittaa muihin samankaltaisiin käsitteisiin, kuten helppokäyttöisyyteen tai saavutettavuuteen. [4, 5] Näin ollen on tärkeää määritellä, mitä käytettävyydellä tässä työssä tarkoitetaan: Käytettävä määritelmä on esitetty International Organization for Standardizationin (ISO) standardissa ISO 9241-11: "*The extent to which a product can be used by specified users to achieve specified goals effectively, efficiency and with satisfaction in a specified context of use*." [1] Käytettävyys siis sisältää käyttäjän subjektiivisen mahdollisuuden käyttää tuotetta tuloksellisesti, tehokkaasti sekä tyydyttävällä tavalla määrätyssä käyttökontekstissa.

Käytettävyys itsessään on tärkeä ominaisuus, koska, ISO:n määritelmän mukaisesti, se käsittelee sitä, kuinka hyödyllinen tuote käyttäjälleen on [1]. Tuote, esimerkiksi tietokoneohjelmisto, jonka käytettävyys on käyttäjiensä mielestä huono, ei todennäköisesti ole hyödyllinen, vaikka se toteuttaisikin hyödyllisiä toimintoja. Huono käytettävyys voi tehdä tuotteen vaikeaksi tai jopa mahdottomaksi käyttää tai hyödyntää, jolloin kyseisen tuotteen käyttäjälleen antama lisäarvo vähenee. Vastaavasti tuotteen hyvä käytettävyys voi lisätä tuotteen arvoa, koska se voi tehdä tuotteen käyttämisestä tai hyödyntämisestä tehokkaampaa. Toisaalta pelkkä käytettävyys ei voi tehdä tuotteesta hyödyllistä jos siinä ei ole hyödyllisiä toimintoja. Toisin sanoen, käytettävyys voi heikentää tai vahvistaa tuotteen tuomaa arvoa, mutta se ei itsessään luo käyttäjälleen konkreettista arvoa. [6]

Käytettävyydeltään laadukkaat tietokoneohjelmistot voivat tarjota ostajaorganisaatiolleen laajemminkin erilaisia hyötyjä. Työtehon lisäämisen lisäksi Macleodin mukaan hyvä käytettävyys vähentää käyttäjän tekemiä virheitä, vähentää työntekijöiden kouluttamisen tarvetta ja tekee ohjelmiston käyttäjistä tyytyväisempiä. Samalla ohjelmiston käyttö tarvitsee vähemmän tukea eikä ohjelmiston dokumentoinnin tarvitse olla yhtä kattavaa kuin käytettävyydeltään huonoilla ohjelmistoilla. [2] Nämä käytettävyydestä saatavat hyödyt parantavat työtehokkuutta sekä vähentävät ohjelmiston käytöstä koituvia kuluja.

Näin ollen ohjelmisto, joka tarjoaa näitä etuja, on oletettavasti kilpailukykyisempi, ja ohjelmistokehittäjien olisi kannattavaa panostaa käytettävyyden kehittämiseen tuotteissaan.

Ominaisuutena käytettävyydellä on merkittävä tausta kognitiivisessa psykologiassa. Kognitiivinen psykologia käsittelee muun muassa ihmisen havainnointikykyä, ajattelemista ja päättelemistä, ja käytettäessä tietokoneohjelmia ihminen käyttää näitä taitoja. Näin ollen ihmisen toiminnan ymmärtäminen näiltä osin voi auttaa myös käytettävyyden ymmärtämisessä. Kognitiivista psykologiaa ja käytettävyyttä yhdistäviä malleja on esitetty erilaisia. Esimerkiksi Miller et al. ovat esittäneet TOTE-mallin (Test, Operate, Test, Exit), joka kuvaa sitä, miten käyttäjä käyttää tietokoneohjelmaa kognitiivisen psykologian näkökulmasta. TOTE-mallin mukaan käyttäjä selvittää ensin, onko ohjelma halutussa tilassa (Test). Mikäli näin ei ole, käyttäjä suorittaa toiminnon, jonka tavoitteena on saada ohjelma haluttuun tilaan (Operate). Tämän toiminnon jälkeen käyttäjä selvittää, oliko operaatio onnistunut, ja jos ei ollut, suoritetaan uusi operaatio (Test). Kun haluttu tila on lopulta saavutettu, käyttö lopetetaan tai aletaan siirtyä seuraavaan haluttuun tilaan (Exit). [7] Sekä käytettävyyden subjektiivisuus että sen tausta ihmistieteisiin lukeutuvassa kognitiivisessa pyskologiassa on hyvä ottaa huomioon, kun käsitellään käytettävyyttä tarkemmin. Näistä seikoista johtuen käytettävyyden insinöörimäinen tarkastelu, jossa ongelmat ovat ratkaistavissa tarkalla suunnittelulla, on ongelmallinen. Sen sijaan, että käytettävyydelle yritetäänkin löytää yleismaailmallista ratkaisua, tulisi sitä käytännössä tarkastella tapauskohtaisesti tuotteiden loppukäyttäjien kanssa.

Shneiderman on kirjoittanut yleisestä käytettävyydestä (eng. *universal usability*), joka käsittelee kaikille käyttäjille yhteistä käytettävyyttä. Tiivistettynä Shneidermanin ajatus yleisestä käytettävyydestä kiteytyy kysymykseen ”Miten informaatio ja kommunikaatiopalvelut voidaan tehdä käytettäviksi kaikille kansalaisille?”[8] Tällaisenaan yleinen käytettävyys lähenee käsitteenä saavutettavuutta ISO:n määritelmän mukaisen käytettävyyden sijaan. Tästä huolimatta yleisen käytettävyyden perusajatusta voidaan käyttää myös pohdittaessa käytettävyyttä: Miten tuotteesta voidaan saada sellainen, että se olisi kaikille käyttäjille yhtä käytettävä, käyttäjän taustasta riippumatta? Vastaavasti voidaan myös pohtia, onko yleinen käytettävyys kyseessä olevan tuotteen kohdalla merkittävä ominaisuus. Tuotteen suunnitteleminen siten, että se olisi kaikkien

potentiaalisten käyttäjien kannalta käytettävä, vaatii usein kompromisseja [8]. Lisäksi on mahdollista, että nämä kompromissit estävät optimaalisimman käytettävyyden toteuttamisen tuotteen ydinkohderyhmälle.

Kuten edellä todettiin, on tuotteen, kuten tietokoneohjelmiston, käytettävyys merkittävä ominaisuus niin tuotteen loppukäyttäjän kuin kehittäjänkin näkökulmasta. Käytettävyys yhdistää eri tieteenaloja kognitiivisen psykologian taustansa johdosta, ja käsitteenä se ei aina ole täysin yksikäsitteinen. Seuraavassa pohditaan käytettävyyden arviointia, jonka kautta voidaan mahdollisesti löytää puutteita tuotteen käytettävyydessä, ja käytettävyyden arvioinnin jälkeen tarkastellaan käytettävyyden kehittämistä paremmaksi.

2.1.1 Käytettävyyden arviointi

Johtuen käytettävyyden kyvystä parantaa tai heikentää tuotetta, on käytettävyyden arvioiminen ja tutkiminen tärkeää. Tuotekehityksen kannalta on tärkeää, että kehitettävällä tuotteella olisi hyvä käytettävyys, jotta se olisi käyttäjilleen mieluisa tuote käyttää, ja jotta se olisi kilpailukykyinen muihin, kilpaileviin tuotteisiin nähden. Ohjelmiston käytettävyyteen voivat vaikuttaa monet erilaiset tekijät, kuten ohjelmistolta saatava palaute, tehtävien vaatima toimintojen määrä ja käyttöliittymän suunnittelu. [9, 10] Yksi tapa käytettävyyden arviointiin on arvioida näitä yksittäisiä käytettävyyteen vaikuttavia asioita sellaisenaan. Esimerkiksi mikäli ohjelmistolla toteutettava tehtävä, joka olisi mahdollista toteuttaa vain muutamalla toiminnolla, vaatiikin kymmeniä toimintoja, voidaan todeta että ohjelmiston käytettävyys ei ole parhaalla mahdollisella tasolla. Toisaalta, tällaisen arvioinnin heikkoutena on se, että se ei ota ohjelmistoa ja sen käytettävyyttä huomioon kokonaisuutena, vaan se keskittyy pelkästään erillisiin yksityiskohtiin.

Yleisesti ottaen käytettävyysarviointia voidaan tehdä joko kvalitatiivisesti tai kvantitatiivisesti. Kvalitatiivisessa arvioinnissa keskitytään arvioimaan käytettävyyden laatua, kun taas kvantitatiivisessa arvioinnissa tutkikaan laskennallisia käytettävyyteen liittyviä tekijöitä, esimerkiksi käyttäjältä tarvittavien toimintojen määriä tai tehtävään tarvittavaa aikaa. Macleod esittää, että pelkkä kvantitatiivinen tarkastelu voi antaa väärää kuvaa ohjelmien käytettävyyden todellisesta tilasta. Esimerkkinä hän mainitsee tilanteen,

jossa kvantitatiivisesti arvioitaisiin ohjelmassa olevien virheiden määrää: Se, että virheitä on vähemmän on toki käytettävyyden kannalta hyvä asia, mutta mikäli ohjelmasta korjataan vain triviaaleja virheitä vakavien virheiden sijaan, antaa kvantitatiivinen arviointi väärän kuvan ohjelman virheistä. [2] Voidaankin todeta, että pelkkä kvantitatiivinen arviointi antaa helposti naiivin kuvan ohjelman käytettävyydestä, ja sen lisäksi tulisikin käyttää myös kvalitatiivisia arviointimetodeita.

Käytettävyysarvioinnissa ongelmana voi olla myös se, että arviointi kohdistuu vääriin asioihin, tai että se ylipäätänsä tehdään väärin. Macleod on esittänyt kolme kysymystä ohjaamaan käytettävyyden arvioimistyötä: Arvioimmeko oikeita asioita, keräämmekö oikeaa dataa ja analysoimmeko dataa oikein? [2] Jotta käytettävyysarvioinnin tulokset olisivat hyödyllisiä, tulisi pohtia, että vastaako tehty arviointi sitä, miten arvioitavaa ohjelmaa tullaan käytännössä käyttämään. Lisäksi arvioinnin aikana on helppo kerätä epäolennaista dataa, jolla ei ole suoranaista vaikutusta käytettävyyteen. Edellä mainittu esimerkki triviaaleista virheistä sopii myös tähän: Niin kauan kun ohjelmassa on käytön kannalta kriittisiä virheitä, on datan kerääminen triviaaleista virheistä epäolennaista. Lopulta, vaikka arviointi vastaisikin reaalimaailman käyttötilanteita ja kerätty data olisi käytettävyyden kannalta olennaista, on arviointi silti turhaa mikäli dataa analysoidaan väärin tai jos analysoinnissa keskitytään epäolennaisiin tekijöihin. Kaiken kaikkiaan käytettävyysarvioinnissa pitäisi olla tarkka sen suhteen miten arviointia suoritetaan. Jotta arviointi varmemmin vastaisi reaalimaailman tarpeita, voitaisiin arvioinnin validoinnissa harkita käytettävän myös arvioitavan ohjelman loppukäyttäjien mielipiteitä. Macleod mainitseekin, että käytettävyys ei riipu pelkästään itse tuotteesta, vaan myös sen käyttäjistä sekä käyttökontekstista. [2] Näin ollen käyttäjien sisällyttäminen arviointiprosessiin voisi tuoda arviointiin näkökulmia, joita varsinaisilla arvioijilla ei välttämättä ole.

Käytettävyyden tutkimiselle, mittaamiselle ja testaamiselle on kirjallisuudessa esitetty useita tapoja. [11, 12, 13] Esimerkiksi Nielsen on esittänyt seitsemän erilaista käytettävyyden tutkimusmetodia. [11] Kuitenkin ISO 9241-11 –standardin määritelmään pohjautuen suuri osa näistä esitetyistä metodeista on ongelmallisia: ISO 9241-11 –standardin mukaan käytettävyys on subjektiivisesti käyttäjän itsensä kokema asia. Esimerkiksi Nielsenin yhtenäisyystutkimus ja standarditutkimus eivät ota huomioon

tuotteen käyttäjän kokemusta käytettävyydestä, vaan ne käsittelevät käytettävyyttä objektiivisesti. Muun muassa standarditutkimuksessa tutkitaan ainoastaan sitä, että vastaako tuotteen käytettävyys vapaasti valittavien käytettävyysstandardien määrityksiä. [11] Koska tämänkaltainen käytettävyyden tutkiminen on tätä työtä varten valitun käytettävyyden määritelmän kannalta ongelmallista, käsitellään tässä työssä käytettävyyden laatua lähtökohtaisesti käyttäjän subjektiivisen käyttökokemuksen pohjalta.

Käytettävyyden arviointia kohtaan on myös esitetty kritiikkiä. Artikkelissaan *Usability Evaluation Considered Harmful (Some of the Time)* Greenberg ja Buxton esittävät kolme tapaa, joilla käytettävyysarvioinnilla voi olla haitallisia vaikutuksia kehitettävälle tietokoneohjelmalle: Greenbergin ja Buxtonin mukaan käytettävyysarvioinnilla pyritään usein perustelemaan suunnitelman olemassaoloa verrattuna verrokkiohjelmiin. Toisin sanoen, sen sijaan että arvioinnilla pyritään kumoamaan suunnitelma, sillä usein yritetään löytää vain muutamia syitä sille miksi kyseinen suunnitelma on muita parempi. Tämä johtaa siihen, että arvioinnin läpikäyneet suunnitelmat ovat heikosti arvioitu, ja niissä piilevät viat voivat jäädä löytymättä. Toiseksi, liian aikainen käytettävyysarviointi voi haitata uusien ohjelmien ja innovaatioiden kehittämistä. Greenbergin ja Buxtonin mukaan sellaisten suunnitelmien, jotka ovat vielä luonnosteluasteella, arviointi voi keskeyttää suunnitelmien kehittämisen kokonaan. Näin siksi, että aikaisessa vaiheessa tällaisissa suunnitelmissa on usein suuriakin käytettävyysongelmia, mutta sen sijaan että kehitystyön alkuvaiheessa arvioitaisiin käytettävyyttä siinä tulisi arvioida itse suunnitelmia. Greenberg ja Buxton esittävätkin, että käytettävyysarviointia tulisi tehdä vasta sen jälkeen, kun ollaan löydetty lopullinen suunnitelma jota lähdetään toteuttamaan. Kolmanneksi, Greenberg ja Buxton esittävät, että arviointi ei tarjoa tietoa innovaatioiden tulevasta hyödyntämisestä niiden tulevassa kulttuurisessa kontekstissa. Toisin sanoen, tänä päivänä tehty käytettävyysarviointi ei pysty ottamaan kantaa siihen, miten arvioitua ohjelmaa tai sovellusta hyödynnettäisiin vuosikymmenten päästä. [14]

Edellä mainittujen käytettävyysarvioinnin haitallisten vaikutusten lisäksi Greenberg ja Buxton väittävät, että käytännössä arviointi on usein liian objektiivista. Heidän mukaansa niin ohjelmistoteollisuudessa kuin akateemisissa piireissä keskitytään usein arvioimaan käytettävyyttä vain objektiivisten metodien avulla. He esittävätkin, että kvantitatiivisten ja

kvalitatiivisten arviointimetodologioiden lisäksi tulisi käyttää myös subjektiivisia arviointimenetelmiä. Tällaisia ovat esimerkiksi asiantuntijoiden mielipiteet ja arvioinnit, ja yhdessä objektiivisten metodien kanssa ne antavat laajemman ja monipuolisemman arviointituloksen. [14] Tämä näkökulma sopii yhteen ISO:n käytettävyyden määritelmän kanssa, joka pohjautuu vahvasti käyttäjän omaan subjektiiviseen kokemukseen tietokoneohjelman käytettävyydestä. Toisin sanoen, käytettäessä ISO:n määritelmän mukaista käytettävyyden käsitettä, pelkkä käytettävyyden objektiivinen arviointi olisi ongelmallista.

Kuten nähtiin, käytettävyyden arvioimiselle on esitetty useita erilaisia tapoja, joiden avulla on mahdollista löytää tietokoneohjelmissa olevia käytettävyysongelmia. Luonteeltaan näitä tapoja on olemassa sekä kvalitatiivisia että kvantitatiivisia ja sekä objektiivisia että subjektiivisia. Kuitenkin käytettävyyden arvioimista kohtaan on esitetty kritiikkiä, ja arviointi voi ollakin haitallista ohjelman kehityksen kannalta mikäli sitä tehdään ohjelmistokehityksen näkökulmasta väärässä vaiheessa tai väärillä metodeilla. [11, 12, 13, 14] Lisäksi objektiivisten arviointimetodien käyttö on jossain määrin ristiriidassa tässä työssä käytettävän käytettävyyden määritelmän mukaan. Toisaalta, tämä ristiriita ei kuitenkaan kumoa sitä mahdollisuutta, että objektiiviset metodit auttaisivat toteuttamaan subjektiivisesti parempaa käytettävyyttä. Voitaisiinkin ajatella, että objektiivisia käytettävyyden arviointimetodeja voidaan hyödyntää kehitettäessä ohjelmaa eteenpäin, mutta lopullisen arvioinnin tulisi pohjautua subjektiivisiin arvioihin.

2.1.2 Käytettävyyden kehittäminen

Erilaisia tapoja, joilla käyttöliittymäsuunnittelussa voidaan tavoitella hyvää käytettävyyttä, on monia, ja käyttöliittymäsuunnittelusta on kirjoitettu useita oppaita ja erilaisten käyttöliittymien käytettävyyttä on tutkittu paljon. [15, 16, 17, 18, 19, 10, 20] Hyödyntämällä näitä olemassa olevia oppeja ja tutkimustuloksia käyttöliittymäsuunnittelusta voidaan suunnitella ja toteuttaa käyttöliittymiä jotka, teoriassa, ovat hyvin käytettäviä. Mitä todennäköisimmin tällaiset käyttöliittymät ovatkin käytettävämpiä kuin käyttöliittymät jotka eivät seuraa näitä oppeja ja tutkimustuloksia, mutta käytettävyys on kuitenkin subjektiivista. Näin ollen käytettävyyssuunnittelua ei tulisi tehdä pelkästään teoreettisen tiedon pohjalta, vaan suunnitelmille pitäisi hakea vahvistusta

ohjelmiston käyttäjiltä. Ivory ja Megraw ovat tutkineet lukuisia Internet-sivustoja jotka käytettävyysasiantuntijat olivat arvioineet käytettävyydeltään laadukkaiksi. He vertasivat näiden Internet-sivustojen ominaisuuksia kirjallisuudessa mainittuihin suosituksiin siitä, millaiset Internet-sivustot ovat hyvin käytettäviä, ja selvittivät miten asiantuntijoiden arviot poikkevat teoreettisista opeista. Tutkimuksessaan he havaitsivat, että useampien mitattujen ominaisuuksien kohdalla oli tutkittujen sivustojen välillä hajontaa, ja selkeitä poikkeuksia kirjallisuudessa esitettyihin suosituksiinkin löytyi. Tämä siis siitä huolimatta, että kyseessä oli käytettävyydeltään laadukkaita Internet-sivustoja. [20] Ivory ja Megraw:n tutkimuksen tulokset osoittavatkin, että pelkkään teoriaan perustuva käytettävyyssuunnittelu ei ole ainoa, tai välttämättä edes paras, tapa luoda käytettävyydeltään laadukkaita ohjelmistoja.

Ivoryn ja Megrawn tutkimuksessa keskityttiin kuitenkin tutkimaan ominaisuuksia jotka olivat helposti laskettavissa, kuten Internet-sivuilla olevien sanojen, kuvien tai värien määrään. [20] Voitaisiinkin olettaa, että mikäli käytettävyyssuunnittelun tukena käytetään teoreettisia suosituksia, tulisi niiden olla laajempia ja mahdollisesti abstraktimpia. Esimerkiksi Shneiderman on esittänyt kahdeksan kultaista sääntöä käytettävyyden parantamiseksi, kuten ”tavoittele yhtenäisyyttä”, ”ota huomioon yleinen käytettävyys”, ”tarjoa hyödyllistä palautetta” ja ”estä virheiden tapahtuminen”. [16] Toisaalta, Shneiderman itsekin toteaa, että nämä säännöt voivat olla ristiriidassa keskenään, ja että ne toimivat enintään lähtökohtana arvioitaessa ohjelmistojen suunnitelmia. [21] Kuitenkin tämänkaltaiset ohjeet antavat enemmän tulkinnanvaraa kuin tarkasti mitattavissa olevat suositukset, ja siten ne voivat olla paremmin sovelletettavissa käytettävyyssuunnittelussa.

Käytettävyyden kehittämistä on käsitelty erilaisista näkökulmista: esimerkiksi Shneiderman on kirjoittanut hauskojen käyttöliittymien suunnittelusta. Artikkelissaan *Designing for Fun: How can we design user interfaces to be more fun?* hän mainitsee erilaisia tapoja, joilla tietokoneohjelmista voidaan saada hauskempia käyttää. Tällaisia tapoja ovat muun muassa animaatioiden käyttö, värit ja ääniefektit. Shneiderman ei kuitenkaan esitä syytä sille miksi ohjelmien tulisi olla hauskoja käyttää, mutta hän kuitenkin muistuttaa, että hauskuuden ei tulisi estää tuottavuutta. [21] Hauskuuden merkitystä käytettävyyden kehittämisessä voisi olla syytä pohtia, sillä voidaan olettaa, että tietokoneohjelma, jonka käyttäminen koetaan hauskaksi, saatetaan myös mieltää

käytettävyydeltään laadukkaaksi. Toisaalta, ohjelman käyttöliittymän hauskuus voi myös olla ominaisuus jonka hyöty käytettävyydelle riippuu ohjelman luonteesta ja käyttäjäryhmästä. Esimerkiksi vapaa-ajan viihdeohjelmissa hauskuuden merkitys voi oletettavasti olla toinen kuin vaikkapa sairaaloissa käytettävissä potilastietojärjestelmissä.

Älykkäät käyttöliittymät ovat käyttöliittymiä jotka sisältävät toimintalogiikkaa, joka suorittaa ohjelman toimintoja tai syöttää ennakoivasti tietoja käyttäjän asemasta käyttäjän aiempien syötteiden perusteella. Jaquero et al. mukaan tällaisten käyttöliittymien on mahdollista parantaa ohjelmien käytettävyyttä, sillä ne vähentävät käyttäjiltä vaadittujen toimenpiteiden määrä. Toisaalta, samalla Jaquero et al. myös toteavat, että jotkut käyttäjät eivät pidä ohjelmista jotka tekevät joskus arvaamattomiakin toimintoja ilman käyttäjältä saatua komentoa. [19] Älykkäitä käyttöliittymiä voidaankin pitää mahdollisena keinona tietokoneohjelman käytettävyyden parantamiseksi, kunhan varmistetaan että ne eivät heikennä käytettävyyttä ohjelman käyttäjien mielestä. Käytännössä älykkäiden käyttöliittymien suunnittelu ja toteuttaminen on vaativaa, sillä se yhdistää useita tieteenaloja kuten ohjelmistotuotantoa, tieteellistä laskentaa tekoälyjen osalta ja ihmistieteitä [19]. Wahlster ja Maybury ovat listanneet älykkäisiin käyttöliittymiin kohdistuvia suurimpia haasteita, joita ovat muun muassa käyttäjän syötteiden analysointi, palautteen generointi, sisällön haku ja käyttöliittymän komponenttien automaattinen asemointi [22]. Tiedostamalla nämä haasteet on mahdollista keskittää älykkäiden käyttöliittymien suunnittelutyötä niihin osa-alueisiin, jotka tulevat suunnittelun ja toteutuksen aikana todennäköisesti vaatimaan eniten resursseja.

Käytettävyyden kehittämistä voidaan tarkastella myös kognitiivisen psykologian avulla. Esimerkiksi Miller et al. esittämää TOTE-mallia voidaan hyödyntää löytämällä siitä käytettävyyden kannalta kriittisimmät vaiheet ja pohtimalla miten näistä vaiheista voidaan tehdä käytettävyydeltään parempia. TOTE-mallissahan käyttäjä sekä evaluoi ohjelman tilaa että suorittaa toimintoja. TOTEn kriittisimmät vaiheet voitaisiin olettaa olevan mallin ensimmäinen Test- sekä Operate-vaiheet: Test-vaiheessa käyttäjän tulisi pystyä mahdollisimman helposti toteamaan onko ohjelma halutussa tilassa, ja Operate-vaiheessa oikeiden toimintojen suorittamisen tulisi olla mahdollisimman helppoa ja tehokasta. Lisäksi väärien toimintojen suorittamisen tulisi olla vaikeaa tai mahdotonta, sillä TOTE-

mallissa väärät toiminnot aiheuttavat rekursiota Operate- ja Test-vaiheiden kohdalla. [7] Tiedostamalla se, miten käyttäjä käyttää ohjelmaa kognitiivisen psykologian näkökulmasta, voidaan käytettävyyden kehittämisessä keskittyä kriittisimpiin toimintoihin. TOTE-mallin valossa tämä tarkoittaisi sitä, että käytettävyyssuunnittelussa pyrittäisiin tekemään ohjelman tilan hahmottamisesta mahdollisimman helppoa, ja että haluttuun tilaan siirtyminen olisi myös tehokasta.

Kuten nähtiin, on käytettävyyden kehittämiselle olemassa hyvinkin erilaisia keinoja ja lähtökohtia. Nimenomaan keinoja käytettävyyden kehittämiselle ovat esimerkiksi Shneidermanin kahdeksan kultaista sääntöä sekä älykkäät käyttöliittymät, sillä ne tarjoavat valmiit työkalut joilla käytettävyyttä voitaisiin mahdollisesti parantaa. [16, 19] Lähtökohtia käytettävyyden kehittämiselle ovat esimerkiksi tarkastelu kognitiivisen psykologian tai käyttöliittymien hauskuuden kautta: Nämä eivät suoranaisesti kerro miten ohjelmasta voidaan tehdä käytettävämpiä, vaan ne ohjaavat käytettävyys- ja käyttöliittymäsuunnittelua tiettyyn suuntaan, joka saattaa auttaa käytettävyyden parantamisessa [7, 21]. Olennaisinta kuitenkin on huomata se, että vaikka käytettävyyden kehittämiselle on esitetty keinoja, ei niitä voida pitää täydellisesti varmoina tapoina käytettävyyden parantamiseksi. Tästä huolimatta näiden keinojen olemassaolo kannattaa tiedostaa, ja niitä kannattaa evaluoida tehtäessä käytettävyyssuunnittelua, mikäli ne sattuisivatkin olemaan juuri kyseisessä tapauksessa hyödyllisiä.

2.2 Sisällönhallintajärjestelmät ja yhteistyöalustat

Sisällönhallintajärjestelmät ovat ohjelmistoja, joiden avulla voidaan, kyseessä olevasta ohjelmistosta riippuen, luoda, ylläpitää, julkaista ja jakaa erilaista digitaalista sisältöä. Boiko määrittelee sisällön dataa ja informaatiota yhdistäväksi kompromissiksi: Sisällön varsinainen ydin on informaatiota, jonka muotoa ei ole rajattu, kuten dokumentteja, taulukoita, kuvia tai ääntä. Mutta koska informaation käsitteleminen on tietokoneille haastavaa, liitetään siihen dataa metatietojen muodossa, jonka avulla saadaan muodostettua sisältöä. Käyttötarkoituksia sisällönhallintajärjestelmillä on useita: Niitä voidaan käyttää niin Internet-sivustojen toteuttamiseen kuin dokumenttien hallintaan ja arkistointiin sekä informaation saatavuuden lisäämiseen yrityksissä ja organisaatioissa. Erilaisista

käyttötavoista johtuen sisällönhallintajärjestelmien toteutustapoja on useita, ja myös eriäviä mielipiteitä siitä, että minkälaiset ohjelmistot ovat varsinaisia sisällönhallintajärjestelmiä, on esitetty. Esimerkiksi joidenkin tahojen mukaan ohjelmistot, joita käytetään pääasiassa Internet-sivustojen toteutukseen, eivät olisi sisällönhallintajärjestelmiä, vaikka ne sisältäisivätkin sisällönhallintajärjestelmien ominaisuuksia. Näiden eriävien mielipiteiden kohdalla kyse on yleensä semanttisista eroista: Joidenkin mielestä vain ohjelmistot, joita käytetään yritysten tai organisaatioiden dokumenttien hallinnointiin olisivat sisällönhallintajärjestelmiä, vaikkakin määritelmänsä mukaisesti myös Internet-sivustojen sisällönhallintajärjestelmät hallinnoivat sivustolla esitettävää sisältöä. [23]

Sisällönhallintajärjestelmää voidaan tarkastella myös sisällönhallintaprosessin kautta. Tällöin sisällönhallintajärjestelmät ovat vain ohjelmistoja, jotka mahdollistavat olemassa olevien sisällönhallintaprosessien toteuttamisen digitaalisesti. Boiko esittää, että yritysten ja organisaatioiden näkökulmasta sisällönhallintaan sisältyy viisi ominaisuutta: Se jakaa liikearvoa, tasapainottaa organisaation sisäisiä voimia, yhdistää sisältöön liittyviä sääntöjä, kerää, hallinnoi ja julkaisee informaatiota sekä toimii teknisenä infrastruktuurina. [23] Näin sisällönhallintajärjestelmän voidaan myös ajatella olevan ohjelmisto, joka mahdollistaa näiden ominaisuuksien toteutumisen.

Yleisesti ottaen sisällönhallintajärjestelmien käytölle on olemassa useita perusteita. Ne helpottavat tuotetun sisällön löytämistä, kun tämä sisältö tallennetaan keskitettyyn paikkaan. Samalla sisältöön kuuluvat metatiedot helpottavat sen löytämistä entisestään, kun tallennettua sisältöä voidaan hakea tai selata sen ominaisuuksien perusteella. Nämä samat ominaisuudet myös mahdollistavat tietotulvan hallinnoimisen sisällönhallintajärjestelmien avulla, kun liikaa informaatiota voidaan hallinnoida metatietojen avulla. [23] Sisällönhallintajärjestelmän käytön vastakohtana voidaankin kuvitella tilannetta, jossa yrityksessä tuotetut dokumentit tallennetaan tietokoneille satunnaisiin paikkoihin ilman erillisiä tunnistetietoja. Tällaisessa tilanteessa pelkästään dokumenttien löytäminen olisi aikaa vievä prosessi, mutta sen lisäksi jotta löydettäisiin oikeasti tarvittava dokumentti, tulee kaikki löydetyt dokumentit lukea läpi. Tämä siksi, että vain siten voitaisiin nähdä mikä dokumentti käsittelee haluttua aihetta.

Sisällönhallintajärjestelmässä tämä tieto voitaisiin tallentaa metatietona, jolloin juuri oikean dokumentin löytäminen olisi helppoa.

Sisällönhallintajärjestelmät, joita käytetään sisällön julkaisemiseen, mahdollistavat sen, että sisällöntuottajat voivat keskittyä työstämään sisältöä sen ulkoasun sijaan. Tämä johtuu siitä, että sisällönhallintajärjestelmiin määritellään erikseen miten julkaistava sisältö esitetään, eikä sisällön esitystapa riipu sisällöstä itsestään. Useissa sisällönhallintajärjestelmissä on toteutettuna myös versionhallintajärjestelmä, joka voi pitää kirjaa sisältöön tehtävistä muutoksista ja mahdollisesti tallettaa kaikki sisällöstä olevat versiot. Versionhallinnasta johtuen järjestelmän käyttäjien on mahdollista tarkistaa miten, milloin ja kuka sisältöä on muokannut, ja sisältö voidaan tarpeen vaatiessa palauttaa aikaisempaan versioon. Muita sisällönhallintajärjestelmille tyypillisiä ominaisuuksia ovat käyttöoikeuksien hallinta ja autentikointi sekä ilmoitukset tapahtumista, kuten automatisoidut sähköposti-ilmoitukset muokatusta sisällöstä. [23, 24]

Varsinkin yrityksissä käytettävien sisällönhallintajärjestelmien haasteeksi on koettu joidenkin järjestelmien monoliittinen rakenne: Tällaiset sisällönhallintajärjestelmät ovat itsenäisiä, ja niistä puuttuu mahdollisuus integroida järjestelmät toisiin, jo ennalta olemassa olemiin järjestelmiin, jotka joko tuottavat tai käsittelevät sisältöä. [25] Yrityksissä voidaan kokea haittana se, että sisällönhallintajärjestelmät eivät täten voi käyttää olemassa olevia resursseja. Samalla näiden järjestelmien käyttö vaatisi enemmän resursseja työntekijöiltä, jotta muissa järjestelmissä tuotettu sisältö saadaan tuotua myös sisällönhallintajärjestelmään. Hankittaessa uutta sisällönhallintajärjestelmää tulisikin ottaa huomioon siltä vaadittavat integroimisominaisuudet, jotta voidaan hankkia tarpeeseen sopiva sisällönhallintajärjestelmä. Integroimisominaisuuksien kohdalla tulisi ottaa huomioon myös se, että minkälaista osaamista niiden käyttö vaatii: Esimerkiksi avoimen lähdekoodin sisällönhallintajärjestelmät lienevät integroitavissa muihin tietojärjestelmiin ohjelmoinnin avulla. Mutta mikäli organisaatiossa olisi tarve sille, että järjestelmän loppukäyttäjät pystyvät hallinnoimaan integrointia, ei tällainen integrointitapa ole tarpeen mukainen.

Sisällönhallintajärjestelmät voidaan karkeasti jakaa kahteen ryhmään, kun tarkastellaan

markkinoilta löytyviä valmiita tuotteita: Yrityskäyttöön suunnattuihin sisällönhallintajärjestelmiin (eng. *Enterprise Content Management System*) sekä ei-yrityskäyttöön suunnattuihin sisällönhallintajärjestelmiin. Yritysten sisällönhallintajärjestelmät voidaan määritellä järjestelminä, jotka voivat hallita kaikkia yrityksen sisältöön liittyviä tarpeita [23]. Vaikka, johtuen sisällönhallintajärjestelmien monimuotoisuudesta sekä yritysten erilaisista tarpeista, tällainen jako ei ole yksiselitteinen, se voi kuitenkin auttaa hahmottamaan sitä, millaisiin käyttötarkoituksiin kukin sisällönhallintajärjestelmä on suunniteltu. Toinen esitetty määritelmä yrityksen sisällönhallintajärjestelmälle on se, että ne ovat organisaation sisällä tai usean organisaation välillä käytettäviä sisällönhallintajärjestelmiä [26]. Tämäkään määritelmä ei ole täysin ongelmaton, sillä miten määritellään organisaation rajat puhuttaessa ohjelmistojen käytöstä? Lasketaanko, esimerkiksi, yrityksen ulkoisten Internet-sivujen käyttö organisaation sisäiseksi käytöksi vai ei?

Yleisesti ottaen, yrityskäyttöön suunnatuiksi sisällönhallintajärjestelmiksi voidaan laskea esimerkiksi ECM Corporationin Documentum [27], Laserfichen Laserfiche Avante [28], OpenText Corporationin OpenText ECM Suite [29] sekä Microsoft Corporationin SharePoint [30]. Muiden sisällönhallintajärjestelmien joukkoon kuuluu esimerkiksi avoimen lähdekoodin Drupal [31] ja Joomla [32] sekä Northwoods Softwaren Titan CMS [33]. Edellä esitelty jako erilaisten sisällönhallintajärjestelmien välillä liittyy enemmänkin ohjelmistojen sisältämiin ominaisuuksiin kuin siihen, mihin tarkoituksiin niitä käytetään tai millaiset tahot niitä käyttävät. Esimerkiksi Microsoftin SharePoint-ohjelmistosta on saatavilla ilmainen laitos, jota voidaan periaatteessa käyttää henkilökohtaisten Internet-sivustojen toteuttamiseen [34]. Vastaavasti esimerkiksi avoimen lähdekoodin Drupalia voidaan teoriassa käyttää mihin tarkoitukseen tahansa sen muokattavuuden vuoksi [31]. Lisäksi Northwood Softwaren Titan CMS:ää on käytetty yritysten intranettien toteutukseen, jolloin sitä voidaan todeta käytettävän organisaation sisällä, vaikka se ei tuotteena sisälläkään ominaisuuksia joilla välttämättä voitaisiin hallita yrityksen kaikkia tarpeita [33]. Näin ollen Titan CMS olisi yhden edellä mainitun määritelmän mukaan yrityksen sisällönhallintajärjestelmä [26], ja toisen määritelmän [23] mukaan se taasen ei olisi yrityksen sisällönhallintajärjestelmä.

Yhteistyöalustat (eng. *collaboration platform*) ovat ohjelmistoja, jotka mahdollistavat tai edesauttavat yhteistyön tekemistä tietokoneilla. Niiden ominaisuuksiin kuuluu muun muassa kommunikointiominaisuuksia, kuten keskustelupalstoja tai reaaliaikaisia pikaviestinsovelluksia, sekä mahdollisuuksia käsitellä dokumentteja yhtä aikaa ja jakaa tietoa rajatun joukon kesken. [35] Jotkut yhteistyöalustat myös integroituvat toimisto-ohjelmistoihin niin, että niiden kautta voidaan avata ja muokata dokumentteja toimisto-ohjelmissa ja tallentaa dokumentteihin tehdyt muutokset suoraan yhteistyöalustassa olevaan tiedostoon. Yhteistyöalusta voidaan myös määritellä ohjelmistona, joka yhdistää yksittäiseen projektiin kuuluvan työn useilta erillisiltä työasemilta ja käyttäjiltä. [36]

Joidenkin yhteistyöalustojen kyky tallentaa tiedostoja tai muuta sisältöä muistuttaa sisällönhallintajärjestelmien ominaisuuksia. Itse asiassa tämänkaltaisia yhteistyöalustoja onkin mahdollista käyttää sisällönhallintaan, vaikka niiden kohderyhmä on usein rajattu, johtuen yhteistyön rajatuista osallistujista. Vastaavasti joitakin sisällönhallintaan tarkoitettuja järjestelmiä voidaan käyttää yhteistyöalustoina hyödyntämällä niiden tiedostojen hallinnan ominaisuuksia. Yhteistyöalustojen ja sisällönhallintajärjestelmien samankaltaisuuksista johtuen jotkut ohjelmistot yhdistävätkin näitä ohjelmistotyyppejä: Esimerkiksi Microsoftin kehittämä SharePoint on juuri tällainen ohjelmisto, joka sisältää sekä sisällönhallinnan että yhteistyöalustan ominaisuuksia, ja sitä voidaan käyttää molempiin tarkoituksiin. [30]

Yhteistyöalustat ovat kuitenkin vain mahdollistavia tuotteita: Terracina et al. mukaan jotta yhteistyöalustoista saadaan käyttöorganisaatiossa esiin hyötyjä, tulee organisaatiossa olla olemassa yhteistyöalustaa hyödyntävän yhteistyön kulttuuri [37]. Näin ollen yhteistyöalustoja ei voida suoranaisesti pitää ratkaisuna yhteistyön parantamiseen. Vastaavasti yhteistyön tehokkuus ei ole riippuvainen pelkästään käytettävän yhteistyöalustan työkaluista ja ominaisuuksista, vaan siitä miten näitä ominaisuuksia hyödynnetään [37]. Tämän takia verrattaessa yhteistyöalustoja toisiinsa tulee yhteistyöalustojen teknisten ominaisuuksien lisäksi huomioida myös se, miten hyvin kyseiset alustat soveltuvat kohdeorganisaatioidensa toimintakulttuureihin.

Johtuen organisaation kulttuurin suuresta merkityksestä yhteistyöalustan käytölle, on myös

yhteistyöalustoina käytettävien ohjelmistojen kirjo laaja. Osa näistä ohjelmistoista on myös sellaisia, joita ei ole edes suunniteltu varsinaisiksi yhteistyöalustoiksi, mutta joita voidaan sellaisina käyttää. Esimerkiksi wiki-sivustoja, joissa sivuston käyttäjien on mahdollista muokata ja tuottaa sivustossa olevaa sisältöä yhdessä muiden käyttäjien kanssa, on käytetty yhteistyöalustoina tiedon levittämiseen ja yhteistyön mahdollistamiseen wikiin tallennetun sisällön suhteen [35]. Lisäksi suurilla organisaatioilla voi olla käytössään useita erillisiä yhteistyöalustoja erilaisiin tarpeisiin [38]. Kommunikointiominaisuuksiensa ansiosta erilaisia sosiaalisen median ratkaisuja, kuten blogeja ja Twitteriä, on myös käytetty yhteistyön edistämiseen, ja niillä on todettu olevan myönteisiä vaikutuksia yhteistyön kannalta [39, 40, 41].

Erityisesti yhteistyöalustana kehitettyjä ohjelmistotuotteita ovat esimerkiksi IBM:n Lotus Notes [42], Oracle Corporationin Oracle WebCenter [43], Microsoftin SharePoint [30] sekä avoimen lähdekoodin Liferay [44]. Näiden tuotteiden tarjoamissa ominaisuuksissa on paljon eroavaisuuksia, vaikka ne lukeutuvatkin samaan yhteistyöalustojen ryhmään. Esimerkiksi Lotus Notes ja Oracle WebCenter tarjoavat muun muassa sähköposti- ja pikaviestinsovelluksia [42, 43], kun taas SharePointissa tällaisia ominaisuuksia ei ole [34]. Vastaavasti Oracle WebCenter, SharePoint ja Liferay tarjoavat myös yrityskäyttöön suunnattuja sisällönhallinnan ominaisuuksia [43, 30, 44], kun taasen Lotus Notesissa tällaisia ominaisuuksia ei ole [42]. Merkittävin yhtäläisyys näiden esimerkkiohjelmistojen ominaisuuksissa lieneekin siinä, että ne kaikki tarjoavat jonkinlaisia kommunikointityökaluja. Lisäksi niitä kaikkia voidaan myös kustomoida joko loppukäyttäjän tai järjestelmän ylläpitäjän toimesta tai niihin voidaan luoda kokonaan uutta toiminnallisuutta ohjelmoimalla. [42, 43, 44, 34]

Edellämainittujen yhteistyöalustojen lisäksi on olemassa myös pilvipalveluina toteutettuja yhteistyöalustoja. Tällainen on esimerkiksi Googlen Google Apps, joka sisältää useita erilaisia ominaisuuksia, mukaan lukien sähköpostin, kalenterin, toimisto-ohjelmiston ja verkkosivustoja. [45] Vastaavanlainen tuote on Microsoftin tarjoama Office 365, joka sisältää sähköpostin, toimisto-ohjelmiston sekä SharePoint Online-ohjelmiston. SharePoint Online on pilvipalveluna toteutettu laitos SharePoint-ohjelmistosta. [46]

Ottaen huomioon yhteistyöalustana käytettävien ohjelmistojen laajan kirjon ja erot näiden ohjelmistojen ominaisuuksissa, yhteistyöalustan valintaa voitaisiinkin lähestyä pohtimalla sitä, mitä kohdeorganisaation toimintakulttuurin osa-aluetta halutaan muuttaa tai käyttää hyödyksi yhteistyöalustojen avulla. Esimerkiksi, mikäli organisaation ongelmana on kommunikaation työkalujen puute, yhteistyöalustaksi saattaa riittää jokin sellainen sosiaalisen median ratkaisu, joka tarjoaisi tarvittavia työkaluja. Toisaalta, jos organisaatiossa halutaan parantaa dokumenttien työstämistä yhdessä, tulisi käytettävällä yhteistyöalustalla olla tähän tarpeeseen soveltuvia ominaisuuksia. Lisäksi yhteistyöalustan valinnassa tulisi ottaa huomioon miten hyvin se sopii yhteen organisaation prosessien ja työkulttuurin kanssa. Myös ohjelmiston muokattavuus voi olla kriittinen ominaisuus käytettäessä uutta yhteistyöalustaa, jotta se saadaan vastaamaan organisaation tarpeisiin paremmin. Kaiken kaikkiaan, jonkin ohjelmiston käyttäminen vain siksi, että sillä ollaan saatu hyötyjä jossakin toisessa organisaatiossa, ei liene tarkoituksenmukaista, varsinkin jos kyseinen ohjelmisto ei vastaa olemassa oleviin tarpeisiin ja jos sitä ei ole mahdollista räätälöidä.

2.3 Monipuoliset Internet-sovellukset

Monipuoliset Internet-sovellukset (eng. *rich internet application*, RIA) on Internet-sovelluksiin liitetty termi, jolla kuvataan sovelluksia joilla on interaktiivinen ja itsenäinen käyttöliittymä, joka voi myös sisältää multimediaelementtejä. RIAt ovat merkittäviä, sillä ne tarjoavat vaihtoehdon perinteiselle HTML:ään (HyperText Markup Language) pohjautuvalle Internet-käyttöliittymämallille, jossa Internet-sivuston käyttäjä lähettää komentoja sivuston palvelimelle ja vastaanottaa uusia sivuja palvelimen käsiteltyä käyttäjältä saadut syötteet. Toisin kuin perinteisessä mallissa, RIAt voivat itsenäisesti käsitellä käyttäjän syöttämää tietoa ilman, että käyttökokemusta tarvitsee katkaista ladattaessa uusia sivuja. Tämän lisäksi RIAt tuovat Internet-sovelluksiin mukaan interaktiivisuutta, jota perinteisistä HTML-pohjaisista Internet-sivuista puuttuu. [47]

Monipuolisten Internet-sovellusten kehitystä on ajanut tarve tehdä Internetissä olevasta sisällöstä houkuttelevampaa käyttää multimedian ja interaktiivisuuden avulla. Niitä käytettiinkin alunperin markkinoinnin ja mainonnan välineinä, mutta nykyiset RIAt ovat

kehittyneet tarjoamaan paljon hyödyllisiä ominaisuuksia myös tavalliseen sovelluskehitykseen. Näiden ominaisuuksien, kuten interaktiivisuuden, visuaalisuuden ja tiedonsiirron minimoimisen [47] ansiosta monipuolisten Internet-sovellusten teknologioita käyttämällä on mahdollista kehittää käytettävyydeltään laadukkaampia Internet-sovelluksia. [48] Seuraavassa tarkastellaan monipuolisten Internet-sovellusten kehittämistä sekä niiden käytön vaikutuksia sovellusten käytettävyyteen.

2.3.1 Monipuolisten Internet-sovellusten teknologiat ja kehittäminen

Bozzon on esittänyt jaon monipuolisten Internet-sovellusten teknologioille, jossa ne jaetaan neljään luokkaan: Komentosarjoihin perustuviin, ohjelmalisäkkeisiin perustuviin, selainpohjaisiin sekä Internetiin pohjautuviin työpöytäteknologioihin. Kullakin näistä luokista on omia ominaispiirteitään. Komentosarjoihin perustuvilla teknologioilla sovelluksen toiminnallisuus toteutetaan komentosarjoilla, ja niissä käyttöliittymät on toteutettu HTML:n ja CSS:n (Cascading Style Sheet) yhdistelmänä. Ohjelmalisäkkeisiin perustuvilla teknologioilla sovellukset suoritetaan erillisen ohjelmalisäkkeen toimesta, joka tarjoaa kehittyneemmät ominaisuudet käyttöliittymän toteuttamiseen ja tapahtumien käsittelyyn. Selainpohjaisissa teknologioissa käytetään hyväksi tietyissä Internet-selaimissa olevia natiiveja interaktiivisia ominaisuuksia, joita voidaan hyödyntää deklaratiivisen ohjelmoinnin avulla. Lopuksi, Internetiin pohjautuvilla työpöytäteknologioilla ladataan Internetistä työpöytäsovelluksia jotka suorittavat ohjelman koodin. [47]

Neljä yleisesti RIA:n toteuttamiseen käytössä olevaa teknologiaa ovat JavaScript, Java, Flash sekä Silverlight. Kukin näistä neljästä teknologiasta vastaa hieman erilaisiin tarpeisiin toisiinsa verrattuna. Käyttäen edellä mainittua Bozzonin esittämää RIA-teknologioiden jakoa [47], JavaScript lukeutuu komentosarjoihin perustuviin teknologioihin kun taas Java, Flash sekä Silverlight kuuluvat ohjelmalisäkkeisiin perustuviin teknologioihin. [49, 50, 51, 52]

JavaScriptiä käytetään lisäämään Internet-selaimessa suoritettavaa toiminnallisuutta staattisiin HTML-sivuihin. JavaScript-sovellukset muokkaavat suoraan HTML-sivun rakennetta, eli niillä ei voi luoda käyttöliittymiä joita pelkällä HTML:llä ei voi esittää. Toisaalta, JavaScript-tuki on saatavilla useissa Internet-selaimissa, eikä se tarvitse erillistä

suoritusympäristöä sovellusten suorittamiseen toisin kuin Flash, Java ja Silverlight. Näin ollen JavaScript-sovellukset ovat lähes kaikkien Internetin käyttäjien käytettävissä. [49, 50, 51, 52] AJAX (Asynchronous JavaScript and XML) yhdistää JavaScriptiä ja XML:ää (Extensible Markup Language), mahdollistaen asynkronisen tiedonsiirron palvelimelle JavaScriptiä käyttäen. Näin ollen JavaScriptin avulla voidaan tehdä sulavasti päivittyviä HTML-pohjaisia sovelluksia. [53] HTML:n viimeisin versio HTML5 sisältää myös monipuolisempia tapoja sisällön esittämiseen HTML-sivuilla. Täten käyttäen sekä HTML5:stä että JavaScriptiä on mahdollista toteuttaa JavaScript-sovelluksia, joiden visuaaliset ominaisuudet lähenevät ohjelmalisäkkeisiin perustuvien RIA-teknologioiden ominaisuuksia. [54] Käyttäen sekä AJAXin että HTML5:n mahdollistamia ominaisuuksia yhdessä voidaan luoda toiminnallisuudeltaan hyvinkin monipuolisia JavaScript-sovelluksia

Flash, Java ja Silverlight tarjoavat sovelluksille oman suoritusympäristönsä, joten näillä teknologioilla toteutettujen sovelluksien toiminnot eivät rajoitu siihen, mitä tavallisella HTML-sivulla voidaan esittää. Adoben kehittämä Flash on erityisesti multimediasovelluksiin, kuten animaatioihin ja peleihin, soveltuva teknologia. Flash sisältää myös ohjelmointiominaisuuksia, tosin ne ovat rajalliset verrattuna Javaan tai Silverlightiin. Sun Microsystemsin Java ei valmiiksi sisällä erityisiä ominaisuuksia multimediaa sisältävän RIAn toteuttamiseen, mutta sen ohjelmointiominaisuudet ovat laajemmat ja tehokkaammat kuin Flashillä. Täten Javan avulla voidaan toteuttaa monimutkaisempaa logiikkaa vaativia monipuolisia Internet-sovelluksia, kun taas Flash soveltuu hyvin yksinkertaisempiin sovelluksiin joille visuaalisuus on tärkeässä osassa. Microsoftin Silverlightia voidaan pitää eräänlaisena kompromissina Flashin ja Javan välillä: Se sisältää tehokkaat ohjelmointiominaisuudet, mutta siinä on myös valmiina ominaisuuksia multimedian ja grafiikan esittämiseen. [50, 51, 52]

Monipuolisten Internet-sovellusten toteutukseen käytettäviä kehitystyökaluja on olemassa hyvin erilaisia. Niitä voidaan, valitusta teknologiasta riippuen, toteuttaa perinteisen ohjelmoinnin työkaluilla tai ohjelmilla, jotka muistuttavat enemmänkin kuvankäsittelyyn ja videoeditointiin suunniteltuja ohjelmia. Jotkut työkalut myös yhdistelevät näitä lähestymistapoja. Nämä työkalut voidaan karkeasti jakaa niin sanotusti ohjelmoijan työkaluihin sekä suunnittelijan työkaluihin. Niiden RIA-teknologioiden kohdalla, joille on

saatavilla sekä ohjelmoijan että suunnittelijan työkaluja, on työkalujen monimuotoisuutta perusteltu kehitystyön jakamisella työkalujen välille. Näissä tapauksissa sovelluksen käyttöliittymä ja visuaalinen ilme toteutettaisiin suunnittelijan työkaluilla, joista sovellus siirrettäisiin jatkokehitettäväksi ohjelmoijan työkaluihin. Riippuen sovelluksesta, sekä siinä käytettävästä RIA-teknologiasta, se voitaisiin ehkä toteuttaa kokonaisuudessaan käyttämällä vain yhtä työkalua. Mutta hajauttamalla toteutusprosessi eri työkaluille voidaan kussakin prosessin vaiheessa käyttää kyseiseen vaiheeseen parhaiten sopivia ohjelmistoja. [52, 51]

Ohjelmistojen käyttöliittymien suunnittelu ja toteuttaminen on työläs ja vaikea prosessi, jossa oikeiden teknologioiden ja työkalujen valinnalla on merkittävä rooli tehtävän työn helpottamisessa. Myers ja Rosson ovat tutkineet käyttöliittymien kehitystä osana ohjelmistojen kehitystä, ja havainneet että lähes puolet tutkittujen sovellusten ohjelmakoodista on liittynyt käyttöliittymiin ja että lähes puolet suunnitteluun sekä puolet toteutukseen käytetystä ajasta on kulunut käyttöliittymiä kehitettäessä. Toisaalta, Myersin ja Rossonin tutkimustulokset ovat vuodelta 1992 ja täten melko vanhoja, eikä niitä voida sellaisenaan verrata nykyiseen ohjelmistokehitykseen uusimpien työkalujen kanssa. Kuitenkin Myersin ja Rossonin saamien tuloksien mukaan oikeiden työkalujen sekä käyttöliittymän hallintajärjestelmien (eng. *user interface management system*), jollaisiksi RIA-teknologiat voidaan laskea, valinta helpotti käyttöliittymien toteutusta. [55] Tämän tuloksen voidaan olettaa pätevän vieläkin, ja mitä todennäköisimmin oikeiden työkalujen valinnan merkitys onkin vain lisääntynyt niiden kehitystyön myötä. Näin ollen voidaankin todeta, että oikeiden ja tarpeenmukaisten RIA-teknologioiden ja kehitystyökalujen valinta on merkittävä myös sovellusten kehitystyön helpottamisen ja nopeuttamisen kannalta.

Kuten nähtiin, on monipuolisten Internet-sovellusten kehittämiseen olemassa hyvinkin erilaisia teknologioita ja työkaluja. Vaikka monessa tapauksessa useammalla kuin yhdellä teknologialla olisi mahdollista luoda toteutettavissa oleva sovellus, on oikean RIA-teknologian valinta tärkeää: RIA-teknologiat on kehitetty erilaisista lähtökohdista, ja niiden ominaisuudet ovat hyvin erilaisia. Teknologioiden erot muun muassa suorituskyvyssä tai kyvyssä integroitua olemassa oleviin järjestelmiin voivat vaihdella merkittävästi. Esimerkiksi tässä työssä tullaan tarkemmin käsittelemään Silverlightia, sillä se on tässä

työssä tarkasteltavan SharePoint-ohjelmiston erityisesti tukema RIA-teknologia. Toisaalta, yhtä lailla tässä työssä olisi voitu valita käytettäväksi joko JavaScript, Flash tai Java; niille ei vain tässä nimenomaisessa tapauksessa olisi ollut samanlaista tukea saatavilla kuin mitä Silverlightille on olemassa. [56] Yleisesti ottaen voidaankin sanoa, että RIA-teknologioissa ei ole olemassa yhtä ainoaa jonka valinta olisi aina oikein, vaan käytettävä RIA-teknologia tulisi aina valita tapauskohtaisesti ja tarpeen mukaan.

2.3.2 Monipuoliset Internet-sovellukset ja käytettävyys

Monipuolisten Internet-sovellusten teknologioita hyödyntämällä on teoriassa mahdollista parantaa Internet-sovelluksen käytettävyyttä [57]. Toisaalta, Komischke on myös varoittanut, että RIA-teknologioilla on väärin käytettynä mahdollista myös heikentää sovellusten käytettävyyttä [48]. RIA-teknologiat ovatkin nimen omaan mahdollistavia teknologioita. Ne tarjoavat sovelluskehittäjille uusia ominaisuuksia hyödynnettäväksi, mutta lopullisen sovelluksen käytettävyyden laatu riippuu siitä miten näitä ominaisuuksia hyödynnetään, eikä siitä mitä teknologioita on valittu käytettäväksi. RIA-teknologioiden ominaisuuksien oikeanlaista käyttöä voidaankin pitää merkittävänä haasteena kehitettäessä Internet-sovellusten käytettävyyttä.

Erilaisia tapoja, joilla monipuolisten Internet-sovellusten teknologioita voidaan käyttää parantamaan sovelluksen käytettävyyttä, on kirjallisuudessa esitetty useita. Esimerkiksi käyttämällä ääniefektejä osana käyttöliittymää voidaan käyttäjälle antaa välitöntä palautetta tai kiinnittää käyttäjän huomio sovelluksessa tapahtuneeseen muutokseen [58]. Scott taasen on esittänyt ajatuksen ”mielenkiintoisista hetkistä” (eng. *interesting moments*), joilla tarkoitetaan hetkiä jolloin sovelluksen käyttöliittymässä tapahtuu havaittavia muutoksia. Tällaiset hetket tapahtuvat yleensä palautteena sovellukselta käyttäjälle, ja ne osaltaan viestittävät muutoksista sovelluksessa. Esimerkiksi tilanne, jossa sovelluksen käyttöliittymä validoi käyttäjän antamaa syötettä ja ilmoittaa syötteen olevan virheellinen korostamalla syötteen kentän punaisella värillä, on mielenkiintoinen hetki. RIAlla on mahdollista luoda sekä useita että monipuolisia mielenkiintoisia hetkiä, jotka voivat parantaa sovelluksen käytettävyyttä kertomalla käyttäjälle sovelluksen tilasta välittömästi ja selkeästi. [17]

Kompleksisissa sovelluksissa eräänä käytettävyysongelmana on käyttäjälle esillä olevien syötekenttien suuri määrä. Tällaisessa sovelluksessa käyttäjän voi olla vaikea hahmottaa mitä tietoa ja mihin kenttiin hänen tulee syöttää. Vastaavasti, mikäli käyttäjän antama syöte on epäkelpo, on käytettävyyden kannalta huono asia jos syötteet evaluoidaan vasta kun käyttäjä lähettää ne käsiteltäväksi. Book et al. ovat esittäneet teoreettisen mallin miten RIAlla voidaan käyttäjille esittää vain kulloinkin tarvittavat syötekentät, ja miten käyttäjän antama syöte voidaan evaluoida reaaliajassa: Tällöin käyttäjälle ilmoitettaisiin heti missä syötteissä on virheitä. Book et al. esittivät myös neljä tapahtumaa joiden yhteydessä evaluointeja tehtäisiin: Siirryttäessä syötekentästä toiseen, siirryttäessä sivulta toiselle, tallennettaessa luonnosta lomakkeesta ja lähetettäessä lomaketta. [59] Käyttämällä tämänkaltaisia syötteiden evaluointeja ja syötekenttien piilotuksia voisi olla mahdollista suoraviivaistaa ja nopeuttaa lomakkeiden täyttämistä. Toisaalta, samalla tulisi pitää huoli siitä, että käyttöliittymästä ei tulisi liian ennalta-arvaamaton: Jos ohjelma piilottaa kenttiä käyttäjän mielestä satunnaisesti, voi lopputulos olla sekava ja käytettävyyden kannalta huono.

Älykkäät käyttöliittymät, jotka Jaquero et al. mukaan voivat parantaa ohjelman käytettävyyttä, ovat monimutkaisempi ja niin sanotusti älykkäämpi versio käyttäjän syötteiden evaluoinnista verrattuna Book et al. esittämään tapaan. Älykkäät käyttöliittymät sisältävät varsinaista tekoälyä, joka voi jopa oppia toimimaan tietyin tavoin käyttäjän toimintojen perusteella, kun taas Book et al. esittämässä mallissa käyttöliittymä reagoisi yksinkertaisempien staattisten sääntöjen perusteella. [19, 59] Toisin kuin staattisilla HTML-sivuilla, joilla ei ole varsinaista toiminnallisuutta, RIAlla on mahdollista toteuttaa myös älykkäitä käyttöliittymiä. Älykkäisiin käyttöliittymiin kohdistuu tosin sama ongelma kuin edellä mainittuun syötekenttien piilottamiseen: Joidenkin käyttäjien mielestä ohjelma, joka reagoi käyttäjän toimintoihin voi olla vaikeasti ymmärrettävä, ja täten käytettävyydeltään huono [19]. Voitaisiin myös ajatella, että älykkäiden käyttöliittymien kohdalla riski siitä, että ohjelma on vaikeasti ymmärrettävä, on suurempi kuin käyttöliittymillä jotka reagoivat staattisten sääntöjen perusteella. Tämä siksi, että käyttäjien voisi olettaa olevan helpompi oppia ymmärtämään staattisia sääntöjä, joiden pohjalta käyttöliittymä toimii. Tällaiset säännöt kun toimivat aina samalla tavalla, ja niiden toiminta on täten ennakoitavista. Sen sijaan älykkäitä käyttöliittymiä hyödyntävät ohjelmat

sisältävät tekoälyä, joka saattaa kehittyä käytön yhteydessä, joka puolestaan tekee ohjelmien ennakoinnista vaikeaa. Mahdollinen ratkaisu tähän voisi olla mahdollisuus kytkeä älykäs käyttöliittymä pois päältä siten, että käyttäjä voisi halutessaan käyttää ohjelmaa ilman, että se reagoisi käyttäjän toimintoihin.

Edellä mainituilla tavoilla RIA-teknologioilla on mahdollista kehittää käytettävyyttä myös kognitiivisen psykologian näkökulmasta. Esimerkiksi TOTE-mallin Test- ja Operate-vaiheista [7] on mahdollista tehdä käytettävyydeltään laadukkaampia RIAn avulla. Scottin esittämillä mielenkiintoisilla hetkillä voidaan käyttäjälle viestittää tietoa ohjelman tilasta, joka puolestaan auttaa käyttäjää evaluoimaan ohjelman sen hetkistä tilaa TOTE-mallin Test-vaiheessa. [17] Esittämällä käyttäjälle vain oikeat syötekentät [59] tai hyödyntämällä älykkäitä käyttöliittymiä [19] voidaan puolestaan vaikuttaa TOTE-mallin Operate-vaiheeseen. Tässä tapauksessa tavoitteena olisi helpottaa oikeiden tilasiirtymien tekemistä ohjelman sisällä. Näin ollen, mikäli Internet-sovelluksen käytettävyydessä on parannettavaa kognitiivisen psykologian näkökulmasta, voi olla että toteuttamalla kyseinen sovellus RIA-teknologioilla nämä ongelmat voidaan ratkaista. Se, että miten kukin yksittäinen käytettävyysongelma tulisi RIAn avulla ratkaista on tosin tapauskohtaista, ja riippuu ongelman lisäksi myös sovelluksen käyttäjistä.

Eräs monipuolisten Internet-sovellusten erityispiirre on se, että ne ovat niin sanottuja yhden sivun sovelluksia. Perinteiset Internet-sovellukset on yleensä jaettu useille fyysisille HTML-sivuille, joiden välillä sovelluksen käyttäjä siirtyy tehdäkseen toimintoja. Tämän mallin sijaan monipuoliset Internet-sovellukset sijaitsevat yleensä vain yhdellä Internet-sivulla. Tällaiset sovellukset vastaavat usein, tosin ei aina, ulkoasultaan ja toiminnaltaan työpöytäsovelluksia, tarjoten käyttäjilleen tutun oloisen käyttökokemuksen. Eräitä esimerkkejä tällaisista sovelluksista ovat esimerkiksi Microsoftin Outlook Web Access, joka on saman kaltainen käyttää kuin työpöytäsovellus Outlook, sekä Google Docs, joka muistuttaa tekstinkäsittelyyn tehtyjä työpöytäohjelmistoja. [60] Eli käyttämällä RIA-teknologioita luomaan työpöytäsovelluksia muistuttavia Internet-sovelluksia voi olla mahdollista tehdä ohjelmia, joita uusien käyttäjien on helpompi oppia käyttämään. Tällöin näiden sovellusten käytettävyys olisi oletettavasti parempi, kuin muiden Internet-sovellusten jotka eivät muistuta työpöytäsovelluksia. Toisaalta, työpöytäsovellusten

kopioiminen voi myös heikentää käytettävyyttä, mikäli kopioitavan sovelluksen alkuperäinen käytettävyys on huono. Tällaisessa tilanteessa voisi olla järkevämpää suunnitella ja toteuttaa RIA-sovellus kokonaan alusta, vaikka sovelluksen käyttäjät joutuisivatkin opettelemaan käyttämään kokonaan uudenlaista ohjelmaa.

Vaikka monipuolisilla Internet-sovelluksilla voidaankin parantaa käytettävyyttä, on syytä muistaa käytettävyyden ISO 9241-11 –standardin määritelmä, jonka mukaan käytettävyys on käyttäjän subjektiivisesti kokema asia [1]. Näin ollen pelkästään käyttämällä RIA-teknologioita toteutettaessa sovelluksia ei välttämättä saavuteta parempaa käytettävyyttä. Päinvastoin, käyttämällä RIAn ominaisuuksia väärin on mahdollista heikentää sovelluksen käytettävyyttä huomattavasti mikäli sovelluksen loppukäyttäjät niin kokevat. Esimerkiksi, jos sovelluksen ainoa tapa antaa käyttäjälle palautetta on ääniefektien avulla, on sovelluksen käyttö mahdotonta mikäli sen käyttäjä on kuuro tai mikäli hänellä ei ole äänentoistoon vaadittavia laitteita. Vastaavasti, sovelluksen on mahdollista antaa käyttäjälleen liian paljon palautteita muun muassa mielenkiintoisten hetkien muodossa. Voidaan olettaa, että tällaisessa tapauksessa käyttäjän olisi vaikeaa löytää palautteiden joukosta kaikista tärkeintä ja olennaisinta informaatiota. Tällöin sovelluksen käyttäminen olisi vaikeaa, kun käyttäjä ei pystyisi selvittämään missä tilassa sovellus kulloinkin on.

3 TYÖSSÄ KÄYTETYT TEKNOLOGIAT

3.1 SharePoint

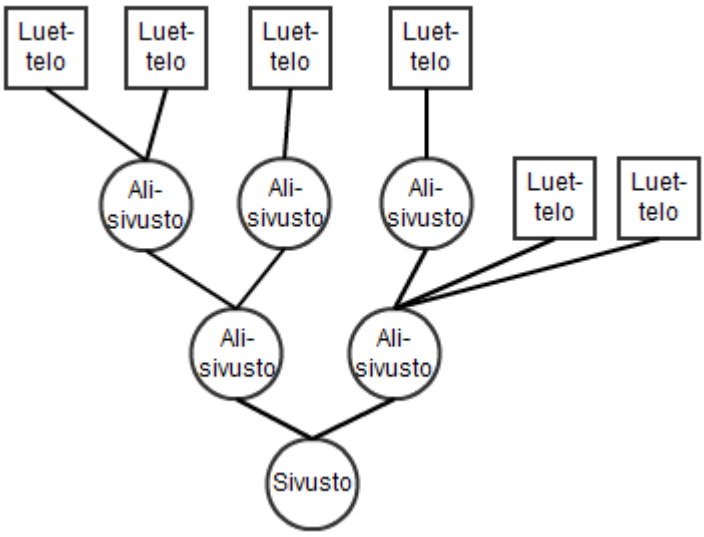
SharePoint on Microsoftin kehittämä ohjelmisto, jonka ensimmäinen versio julkaistiin vuonna 2001 ja nykyinen versio vuonna 2010. Tässä työssä käsitellään pääasiassa vain SharePointin uusinta 2010-versiota. SharePoint 2010:stä on olemassa kolme eri laitosta: SharePoint Foundation 2010 on ilmainen tuote, joka sisältää rajatun alajoukon kaikista SharePointin ominaisuuksista. SharePoint Server 2010 on Foundationia kattavampi, mutta maksullinen, versio, ja SharePoint Enterprise 2010 sisältää kaikki SharePointin ominaisuudet. SharePointia voidaan käyttää useisiin erilaisiin tarkoituksiin, kuten sisällönhallintaan, kommunikointiin, tiedostojen jakeluun sekä kollaborointiin. Käytännössä SharePoint-alustalla luodaan erilaisia Internet- tai intranet-sivustoja, joilla edellä mainittuja toimintoja voidaan tehdä. SharePointin toiminnallisuuksia voidaan myös laajentaa ohjelmoinnin avulla, jolloin SharePointia voidaan pitää myös ohjelmistoalustana. [34] Tässä työssä SharePointia tarkastellaan pääasiassa sisällönhallintajärjestelmänä sekä yhteistyöalustana, jonka toiminnallisuuksia voidaan tarvittaessa laajentaa erillisillä ohjelmistototeutuksilla.

SharePoint voidaan käytännössä asentaa joko yhdelle tietokoneelle, jolloin kyseinen tietokone toimii itsenäisenä palvelimena, tai usealle tietokoneelle, jolloin nämä koneet muodostavat yhdessä SharePoint-farmin. SharePoint-farmissa voi olla kahdenlaisia koneita: Sovelluspalvelimia, jotka suorittavat palvelimella ajettavia binäärejä, sekä HTTP-palvelimia (Hypertext Transfer Protocol), jotka vastaavat HTTP-viestien vastaanottamisesta ja lähettämisestä käyttäjille. Käytön kannalta SharePoint-farmin koolla ja palvelimilla on merkitystä vain suorituskyvyn osalta, mutta toteutettaessa SharePoint-sovelluksia farmin ominaisuuksilla voi olla merkitystä. Esimerkiksi käytettäessä palvelimien välimuisteja tai kovalevyä tiedon tallennukseen tallennettu tieto on saatavissa vain sillä palvelimella, jolla kyseinen tieto tallennettiin. [56] Yksinkertaisuuden vuoksi tässä työssä käsitellään vain yhdelle palvelimelle asennettua SharePointia, koska SharePoint-farmin käyttö ei suoranaisesti vaikuta tässä työssä käsiteltäviin aiheisiin.

3.1.1 SharePoint sisällönhallintajärjestelmänä

SharePoint perustuu puumaiseen sivustorakenteeseen, jossa yhdellä SharePoint-sivustolla voi olla tavallisen sisällön lisäksi useita alisivustoja, ja kullakin alisivustolla voi olla omia alisivustojaan, ja niin edelleen. Kukin näistä alisivustoista on oma erillinen kokonaisuutensa omine käyttäjäoikeuksineen ja sisältöineen. Tämä malli helpottaa SharePointiin tallennetun sisällön hallinnointia, kun se voidaan jakaa loogisten yksiköiden kesken erillisille sivustoille. Näiden sivustojen muodostamaa kokonaisuutta kutsutaan sivustokokoelmaksi. Esimerkiksi SharePoint-sivustokokoelmaan voitaisiin luoda omat sivustonsa kullekin organisaatiossa olevalle tiimille, ja kunkin tiimin käsittelemä sisältö tallennettaisiin aina sen omalle sivustolle. Tällöin tähän sivustokokoelmaan tallennettu sisältö olisi loogisesti jaettuna niiden tiimien mukaan, jorka sisältöä tuottavat. [61]

Edellä mainittuihin SharePoint-sivustoihin tallennettava sisältö lisätään niin sanottuihin luetteloihin (eng. *list*) ja dokumenttikirjastoihin (eng. *document library*). Luettelot ja dokumenttikirjastot ovat SharePointin abstraktioita relaatiotietokantojen käyttämistä tauluista, joihin voidaan luoda kenttiä joihin tallennetaan dataa. Lisäksi luetteloiden riveihin voidaan liittää n-kappaletta liitetiedostoja. Dokumenttikirjastot poikkeavat luetteloista siinä, että jokaisella dokumenttikirjaston rivillä on aina tasan yksi tiedosto, kun taas luettelon rivillä liitetiedostoja on siis 0-n kappaletta. [34] Kuhunkin SharePoint-sivustoon voidaan luoda lukuisia luetteloita ja dokumenttikirjastoja sisällön organisoimiseksi sivuston sisällä. Aiemmin esitetty esimerkki, jossa kullekin organisaation tiimille luotiin oma SharePoint-sivusto sisällön organisoimiseksi, voitaisiin siis toteuttaa myös siten, että kaikki tiimit käyttävät yhteistä sivustoa. Tässä jaetussa sivustossa kullakin tiimillä olisi omat luettelot ja dokumenttikirjastot, joihin tiimien käsittelemä sisältö tallennettaisiin. [61] Se, miten SharePointia käytetään sisällön organisoimiseen riippuneekin enemmän käyttötarpeista. Esimerkiksi pienillä ja suurilla organisaatioilla voi olla hyvinkin erilaiset tarpeet sisällön hallinnalle, johtuen jo pelkästään sisällön määrästä. Kuvassa 1 on esitettynä eräs SharePoint-sivustokokoelma alisivustoineen ja luetteloineen puumaisena rakenteena.



Kuva 1. SharePoint-sivustokokoelman esimerkkirakenne.

Luetteloiden ja dokumenttikirjastojen tiedostot ovat saatavilla kaikille käyttäjille, joille kyseiseen luetteloon, dokumenttikirjastoon tai tiedostoon on annettu tarvittavat käyttäjäoikeudet. Haluttaessa oikeudet voidaan antaa myös anonyymeille käyttäjille, jolloin tiedostot ovat kaikkien saatavilla. Luetteloissa ja dokumenttikirjastoissa tiedostoihin voidaan siis myös liittää metatietoja, sillä kunkin luettelon tai dokumenttikirjaston rivin sarakkeet kohdistuvat juuri kyseisen rivin liitetiedostoihin. Luetteloiden ja dokumenttikirjastojen sisältöä voidaan suodattaa ja järjestellä näiden metatietojen mukaan, jolloin oikeiden tiedostojen löytäminen helpottuu. [34]

SharePoint-sivustossa on mahdollista suorittaa sivustokokoelman laajuisia hakuja, jotka etsivät hakusanoja vastaavia kohteita kaikista sivustokokoelmassa olevista sivustoista ja näiden sivustojen luetteloista ja dokumenttikirjastoista. Käyttämällä tiedostoihin liitettyä metatietoa voidaan SharePointin haun avulla löytää haluttuja tiedostoja tehokkaammin. Lisäksi SharePointin haut osaavat tutkia myös joidenkin tiedostojen sisältöä: Esimerkiksi Microsoft Office –ohjelmiston tuottamien tiedostojen, kuten Microsoft Word- ja Microsoft Excel –tiedostojen, sisällöt käydään haun yhteydessä läpi. [34] Näin ollen mikäli luetteloon tai dokumenttikirjastoon tallennettu Word-dokumentti sisältää annetut hakusanat, palautetaan tämä dokumentti haun tuloksena vaikka dokumenttiin liitetty metatieto ei vastaisikaan hakua.

SharePointin dokumenttikirjastoissa on mahdollista käyttää versionhallintaa. Tällöin SharePoint pitää kirjaa tiedostoihin tehdyistä muutoksista ja säilyttää kaikki tiedostoista olevat versiot. SharePointissa on myös joukko sisällön julkaisemiseen suunnattuja ominaisuuksia. SharePointin tapauksessa nämä julkaisuominaisuudet kohdistuvat pelkästään Internet-sivuihin eikä muuhun SharePointiin mahdollisesti tallennettavaan sisältöön. Mikäli julkaisuominaisuuksia käyttävällä SharePoint-sivustolla luodaan uusi Internet-sivu, tai muokataan jo olemassa olevaa sivua, eivät muutokset näy kaikille käyttäjille ennen kuin sivu on julkaistu. Ennen julkaisua muutokset ovat nähtävissä sivustoa hallinnoiville käyttäjille, ja sivuston asetuksista riippuen julkaisun yhteydessä voidaan käyttää hyväksyntätyönkulkua, jossa muutokset joko hylätään tai hyväksytään julkaistavaksi. [34] Työnkulut ovat ohjelmallisesti toteutettuja ja työntekoa ohjaavia prosesseja. Esimerkiksi SharePointin hyväksyntätyönkulku ilmoittaa hyväksyjän roolissa oleville henkilöille, että työnkulkuun liitetty tiedosto tulisi käsitellä. Kun hyväksyjä on käsitellyt tiedoston työnkulku lähettää tästä tiedon joko seuraavalle hyväksyjälle tai tiedoston luojalle, riippuen työnkulun vaiheesta ja hyväksyjän antamista syötteistä. Näin työnkulku ohjaa prosessia kunnes se on käyty läpi ja hyväksyttävä tiedosto on joko hyväksytty tai hylätty. [61]

Luetteloihin ja dokumenttikirjastoihin voidaan myös liittää sisältöä käsittelevää toimintalogiikkaa. SharePointin mukana tulee joukko valmiita työnkulkuja, joita voidaan liittää luetteloihin sekä dokumenttikirjastoihin. Nämä työnkulut voidaan asettaa ajettaviksi luotaessa tai muokattaessa sisältöä, ja niiden toimintoihin sisältyy esimerkiksi sähköpostihälytysten lähettäminen ja muutosten hyväksyntäprosessien läpikäyminen. Luetteloihin ja dokumenttikirjastoihin on myös mahdollista ohjelmoimalla toteuttaa tapahtumankäsittelijöitä, jotka suoritetaan kun luettelossa tai dokumenttikirjastossa tapahtuu muutoksia. Näiden tapahtumankäsittelijöiden on muun muassa mahdollista estää muutosten tapahtuminen ohjelmallisesti mikäli muutokset ovat virheellisiä. [56] Näiden ominaisuuksien ansiosta SharePointin luettelot ja dokumenttikirjastot eivät ole pelkästään staattisia tietovarastoja, vaan ne osaavat tarvittaessa reagoida tallennettuun sisältöön ja sisältöön kohdistuviin tapahtumiin.

SharePointin sisällönhallintaominaisuuksiin kuuluu myös dokumenttiarkisto.

Dokumenttiarkisto on spesifi SharePoint-sivusto, jonka tarkoituksena on säilyttää sinne siirretyt dokumentit sellaisenaan ja erillään sivustoista joilla sisältöä muokataan. [34] Dokumenttiarkiston käytön mielekkyyttä voidaan kuitenkin siinä mielessä pohtia, että vastaavia toiminnallisuuksia voidaan luoda SharePoint-sivustoilla käyttäen hyväksi tavallisia dokumenttikirjastoja ja arkistoi paikallaan (eng. *manage records in place*) – ominaisuutta [62]. Lopullisesti kyse lienee kuitenkin tapauskohtaisista tarpeista, sillä dokumenttiarkistot voivat olla tarpeellisia joko suurille organisaatioille tai organisaatioille, jotka tuottavat erittäin paljon arkistoitavaa materiaalia. Tällöin erillisen arkistokokonaisuuden käyttö voi olla perusteltua.

Kuten nähtiin, SharePoint sisältää sisällönhallintajärjestelmille ominaisia toimintoja ja ominaisuuksia: Sitä voidaan käyttää keskitettynä paikkana informaation tallennukseen, ja SharePoint liittää tähän informaatioon dataa, jolloin saadaan aikaan sisältöä. Pääsyä sisältöön voidaan rajoittaa asettamalla sisällölle rajoitettuja käyttöoikeuksia, tai vastaavasti sisällöstä voidaan tehdä täysin julkista. Sisältöä voidaan joko selata tai hakea metatietojen perusteella, ja siihen tehtyjä muutoksia voidaan seurata versionhallinnan avulla. Lopulta julkaisuominaisuuksien avulla voidaan hallita sitä, milloin muokatusta sisällöstä tehdään kaikille käyttäjille julkista.

3.1.2 SharePoint yhteistyöalustana

SharePointin puuta muistuttava sivustorakenne mahdollistaa yhteistyön jakamisen useille erillisille sivustoille työtä tekevien loogisten yksiköiden mukaan. Sivustokokoelmaan voidaan, esimerkiksi, luoda kullekin organisaation tiimille tai projektille oma sivustonsa, jonka sisällä työnteko tapahtuu. Tällainen malli mahdollistaa käyttäjäoikeuksien asettamisen sivustoille tiimi- tai projektikohtaisesti, ja se antaa kullekin tiimille tai projektiryhmälle mahdollisuuden hallinnoida omaa sivustoaan erillään muista. [61] Sivustojen puuta muistuttava rakenne ei itsessään edistä varsinaista yhteistyötä, mutta yhteistyön yksiköiden eristämisen omiin sivustoihinsa voidaan ajatella helpottavan yhteistyön hallintaa kokonaisuutena.

Erityisesti yhteistyötä edistäviä ominaisuuksia SharePointissa on erilaisia. Kommunikointiominaisuuksia, joita SharePoint tarjoaa, ovat esimerkiksi keskustelupalstat

ja blogit. Keskustelupalstoja on mahdollista luoda yksittäisiin SharePoint-sivustoihin, jolloin ne toimivat rajatun joukon kommunikaatiovälineinä. SharePointissa luodut blogit voidaan tehdä joko henkilökohtaisiksi, tai niiden ylläpito voidaan jakaa tietyn käyttäjäryhmän kanssa. SharePoint ei itsessään sisällä pikaviestinominaisuuksia, mutta se on integroitu Microsft Lync-pikaviestinsovelluksen kanssa, joka aiemmin tunnettiin nimellä Microsoft Office Communicator [63]. Tämä integraatio mahdollistaa reaaliaikaisen keskustelun aloittamisen SharePoint-sivuston kautta, ja käyttäjät voivat SharePoint-sivustosta käsin nähdä missä tilassa muut pikaviestimen käyttäjät ovat. [34]

SharePoint-sivustoihin voidaan luoda sekä kalentereita että tehtäväluetteloita helpottamaan yhteistyön organisoimista. Kalentereihin voidaan tallentaa tietoja yhteisistä tai henkilökohtaisista tapahtumista, ja niihin tallennettuja tapahtumia voidaan siirtää erilaisiin asiakasohjelmiin kuten Microsoft Outlookiin. Tehtäväluetteloihin voidaan luoda tietyille käyttäjille kohdistettuja tehtäviä, joihin on liitetty tehtävää kuvaavaa dataa. Näitä tehtäviä voidaan joko luoda manuaalisesti, tai sitten niitä voidaan tehdä automaattisesti osana, esimerkiksi, sivuston sisältöön liitettyjä työnkulkuja. Eräs esimerkki tällaisesta työnkulusta on SharePointin hyväksyntätyönkulku, joka voidaan käynnistää automaattisesti kun dokumenttikirjastoon lisätään uusi tiedosto. Työnkulku luo asetustensa mukaiset tehtävät tietyille käyttäjille, ja nämä tehtävät tallennetaan sivustossa olevaan tehtäväluetteloon. SharePointin tehtävät pitävät sisällään tiedon tehtävän tilasta, joten ne eivät pelkästään hyödytä tekijöitään, vaan myös muita sivuston käyttäjiä joiden tarvitsee tietään tehtävien edistymisestä. [61]

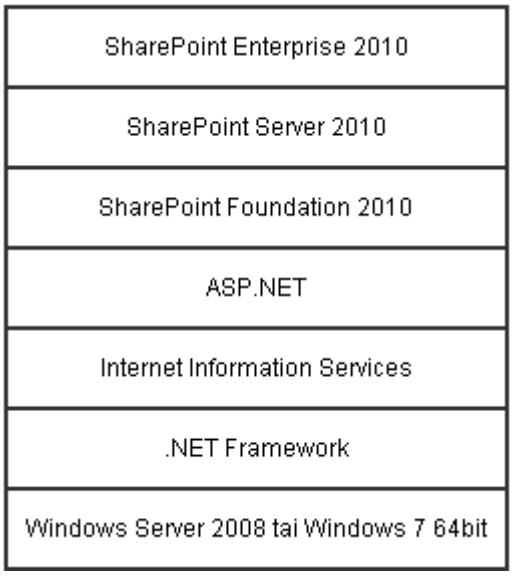
SharePointin avulla on mahdollista tehdä wiki-tyylisiä sivustoja, joissa kaikki kyseisen sivuston käyttäjät voivat yhdessä luoda ja muokata sisältöä. [34] Wiki-sivustojen merkitystä yhteistyön välineenä on tutkittu, ja niiden on todettu mahdollistavan paremman yhteistyön ilmenemistä [35]. SharePoint mahdollistaa myös dokumenttikirjastoihin tallennettujen Microsoft Office-tiedostojen yhtä aikaisen käsittelyn ja muokkaamisen. Se, miten yhtä aikainen tiedostojen muokkaus on SharePointissa toteutettu vaihtelee tiedostotyypin mukaan. Esimerkiksi, muokattaessa Microsoft Word-tiedostoja käyttäjät avaavat dokumenttikirjastoon tallennetun tiedoston, ja näkevät korostettuina mitä kohtia tiedostosta muut käyttäjät ovat muokkaamassa. Samalla nämä työstettävät kohdat lukitaan,

jotta useampi käyttäjä ei pysty tekemään niihin päällekkäisiä muutoksia. Lopuksi, yhden käyttäjän tekemät muutokset näytetään muille käyttäjille heti, kun ne on tallennettu takaisin dokumenttikirjastoon. [34]

Kuten luvussa 2.2 todettiin, ovat yhteistyöalustat vain mahdollistavia tuotteita [37], ja hyvinkin erilaisia ohjelmistoja ja niiden ominaisuuksia voidaan käyttää yhteistyön edistämiseen. Edellä mainitut SharePointin ominaisuudet ovatkin sellaisia, joiden voidaan ajatella olevan osaltaan suunniteltuja juuri yhteistyötä silmällä pitäen: Niitä käytetään joko kommunikointiin, varsinaiseen työntekoon tai työn organisointiin. Kuitenkaan nämä ominaisuudet eivät sulje pois muiden SharePointin ominaisuuksien käyttöä yhteistyön apuvälineinä. Vastaavasti, näiden ominaisuuksien olemassaolo ei ole tae sille, että erityisesti niitä tultaisiin käyttämään yhteistyön edistämiseen SharePointia käyttävissä organisaatioissa.

3.1.3 SharePoint ohjelmistoalustana

SharePoint on toteutettu Microsoftin kehittämää .NET Framework –ohjelmistokehystä käyttäen, ja se hyödyntää ASP.NET-kehystä ja käyttää Internet Information Services -palvelinohjelmistoa (IIS). Käyttöjärjestelmänä SharePoint 2010-palvelimella tulee olla käytössä joko Windows Server 2008 64-bittinen tai 64-bittinen versio Windows 7-käyttöjärjestelmästä tai uudemmasta. Kuvassa 2 on esitettynä SharePointin käyttämät teknologiat. Vastaavasti SharePoint-ohjelmistokehityksessä hyödynnetään näitä samoja teknologioita, ja SharePointin toiminnallisuuksia voidaan myös laajentaa räätälöimällä IIS:n toiminnallisuuksia. Kaiken kaikkiaan SharePoint-ohjelmoinnissa käytettävissä olevia teknologioita on lukuisia, ja niitä käytetään erilaisiin ohjelmistototeutuksiin tilanteen mukaan. Esimerkiksi toteutettaessa SharePoint-palvelimella suoritettavia ohjelmia käytettävät teknologiat ovat erilaiset, kuin toteutettaessa ohjelmia, jotka suoritetaan käyttäjän päätelaitteella. Yleisesti ottaen SharePointia voidaan käyttää ohjelmistoalustana kahdella tapaa. SharePointin olemassa olevia toiminnallisuuksia voidaan laajentaa ja räätälöidä erilaisilla ohjelmointitoteutuksilla, ja toiseksi, SharePointin päälle voidaan toteuttaa kokonaan uusia ohjelmistoja, jotka hyödyntävät SharePointin ominaisuuksia. Näitä tapoja voidaan myös yhdistellä, jolloin saadaan aikaan ohjelmistoja, jotka käyttävät hyväkseen SharePointin laajennettuja toimintoja. [56]



Kuva 2. SharePoint 2010:n käyttämät teknologiat

Kuten todettua, SharePoint-ohjelmoinnissa on käytettävissä useita ohjelmointitekniikoita, ja SharePoint-ohjelmia voidaan tehdä useihin erilaisiin ympäristöihin. Yksi tapa SharePoint-ohjelmoinnissa on luoda laajennuksia tai uusia toiminnallisuuksia SharePoint-sivustolle. Tällöin ohjelmakoodi suoritetaan SharePoint-palvelimella, ja se toteutetaan .NET-ohjelmistokehyksellä käyttäen hyväksi SharePointin omia .NET-luokkakirjastoja. Toinen tapa luoda SharePoint-ohjelmia on tehdä Internet-selaimessa suoritettavia sovelluksia, jotka kommunikoivat SharePoint-palvelimen kanssa asynkronisesti. Tällä tapaa voidaan luoda esimerkiksi interaktiivisia lisätoimintoja SharePoint-sivustolle, ja käytettävissä olevia teknologioita ovat esimerkiksi Javascript ja Silverlight. Kolmas tapa on luoda täysin erillään SharePointista suoritettavia ohjelmia, jotka käyttävät SharePointia tietovarastona, ja niitä voidaan toteuttaa lukuisilla erilaisilla tekniikoilla. [56]

SharePoint käyttää pääasialliseen tiedon tallentamiseen Microsoft SQL Server – tietokantaa, johon tallennetaan sekä osa SharePoint-ympäristön konfiguraatioasetuksista että suuri osa SharePoint-sivuston sisällöstä. Tämän tietokannan lisäksi osa asetuksista ja sisällöstä tallennetaan myös suoraan palvelimella olevalle levyasemalle. Kolmantena tietovarastona SharePoint voi käyttää ulkoisia tietokantoja tai muita järjestelmiä. SharePointin Business Connectivity Services -ominaisuus (BCS) mahdollistaa tiedon lukemisen ja käsittelemisen erillisistä järjestelmistä SharePointin luettelo-abstraktion

välityksellä. Näitä BCS:ä käyttäviä luetteloita kutsutaan ulkoisiksi luetteloiksi, ja niitä voidaan käyttää ohjelmoitaessa SharePoint-sovelluksia samaan tapaan kuin muitakin luetteloita. BCS sisältää mahdollisuuden käyttää muutamia erilaisia tietokantoja suoraan, ja muille tietokantatuotteille on mahdollista toteuttaa BCS:een laajennuksia, jotka toteuttavat tietokantayhteyksissä tarvittavan tiedonsiirron. [56] Peruste BCS:n käytölle on siinä, että jo olemassa olevaa dataa voidaan hyödyntää helposti SharePoint-sovellusten toteutuksessa. Samalla SharePointin loppukäyttäjien on helpompi hallinnoida dataa, kun sitä voidaan muokata SharePointin luettelonäkymien kautta tietokantojen omien hallintaohjelmistojen sijaan.

Pääasialliset SharePoint-ohjelmistokehitykseen käytettävät työkalut ovat Microsoftin toteuttamat Visual Studio 2010 ja SharePoint Designer. Toki muitakin ohjelmia voidaan käyttää, mutta edellä mainitut ovat erityisesti Microsoftin tukemia, ja ne sisältävät erityisesti SharePoint-ohjelmistokehitystä helpottavia ominaisuuksia. Visual Studio 2010:tä käytetään niin sanotusti perinteisempään ohjelmistokehitykseen, jossa toiminnallisuutta toteutetaan ohjelmoimalla. Sillä voidaan myös tehdä useita sellaisia ohjelmistokehitysprojekteja SharePointille, joita SharePoint Designer ei tue. SharePoint Designerilla luodaan toiminnallisuuksia SharePointiin erilaisten helppokäyttöisten työkalujen kanssa ilman varsinaista ohjelmointia. Esimerkiksi työnkulkujen toteuttaminen SharePoint Designerilla tapahtuu vain asettamalla työnkulun tapahtumia paikoilleen hiiren avulla, kun taas Visual Studiolla työnkulkujen toteuttaminen vaati useasti ohjelmointia. [56]

SharePoint Designerilla on kuitenkin kolme selvää heikkoutta Visual Studioon nähden: Ensinnäkin, ohjelmointituen puuttumisen takia sillä ei voida toteuttaa kaikkia niitä ominaisuuksia, mitä Visual Studiolla voidaan toteuttaa. Toiseksi, SharePoint Designer tallentaa tehdyt muutokset suoraan olemassa olevaan SharePoint-sivustoon, toisin kuin Visual Studio, joka tallentaa muutokset erillisiin projektitiedostoihin. Tästä johtuen SharePoint Designerilla tehdyt toteutukset ovat helposti sivustokohtaisia, kun taas Visual Studiolla toteutettuja projekteja voidaan helposti asentaa useisiin SharePoint-sivustoihin. SharePoint Designerilla sivustoihin luodut ominaisuudet voidaan tosin jälkeenpäin tallentaa tiedostoihin, jotka voidaan ladata myös muihinkin SharePoint-sivustoihin.

Kuitenkin projektinhallinnan näkökulmasta on parempi, että toteutettavaan ratkaisuun tehtävät muutokset eivät vaikuta projektin kehitysympäristöön jo toteutusvaiheessa, kuten SharePoint Designeria käytettäessä tapahtuu. Kolmanneksi, SharePoint Designerista puuttuu versionhallinnan tuki, joten projektiin tehtäviä muutoksia on vaikea hallinnoida. Visual Studiossa taasen on sisäänrakennettuna Team Foundation Server-versionhallinta, jonka avulla projektin ylläpitäminen on helpompaa. [56] Johtuen SharePoint Designerin puutteista projektinhallinnan näkökulmasta voi sen käyttöön SharePoint-ohjelmistokehityksessä suhtautua kriittisesti. Ehkäpä merkittävin SharePoint Designerin etu on sen käytön helppous ohjelmointia osaamattomille käyttäjille, joten sitä voidaankin pohtia käytettäväksi vain silloin, kun tarpeena on luoda pieniä muutoksia SharePoint-ympäristöön ilman ohjelmointia.

SharePoint-palvelimen ulkopuolella suoritettavat SharePoint-sovellukset käyttävät kommunikointiin WWW-sovelluspalveluita, jotka on toteutettu joko Windows Communication Framework:lla (WCF), ASP.NET-palveluina tai Representational State Transfer:lla (REST). Silverlight-, Javascript- ja .NET-sovelluksille on tarjolla myös SharePointin oma asiakassovellusten luokkakirjasto, joka tarjoaa SharePointin sisältämien WWW-sovelluspalveluiden toiminnallisuuksia helpommin käsiteltävässä muodossa. Muilla teknologioilla toteutettujen sovellusten tulee käyttää näitä WWW-sovelluspalveluita suoraan ilman tätä luokkakirjastoa. SharePoint-ohjelmoijan on myös mahdollista luoda omia SharePoint-WWW-sovelluspalveluita, mikäli olemassa olevat WWW-palvelut eivät vastaa toteutettavan ohjelmiston tarpeisiin. [56]

SharePoint-ohjelmistojen toteutusta voidaan lähtökohtaisesti verrata ohjelmistojen toteuttamiseen muilla Internet-sivustojen toteuttamiseen tarkoitetuilla alustoilla ja tekniikoilla, kuten Drupalilla [31], Joomlalla [32] tai ASP.NETillä [64], ja pohtia sitä, miten SharePoint-ohjelmointi eroaa muista saatavilla olevista vaihtoehdoista. SharePoint-ohjelmoinnissa voidaan ajatella olevan kaksi merkittävää tekijää ohjelmiston toteutuksen kannalta. Siinä voidaan käyttää hyväksi SharePointissa olevia abstraktioita kuten luetteloita ja dokumenttikirjastoja, jotka vastaavat relaatiotietokantojen tauluja. Toisaalta, SharePoint myös sisältää sisäänrakennettuna lukuisia erilaisia ominaisuuksia, jotka muutoin ehkä pitäisi ohjelmistoon erikseen toteuttaa. Esimerkiksi SharePointin valmiiksi toteutettu

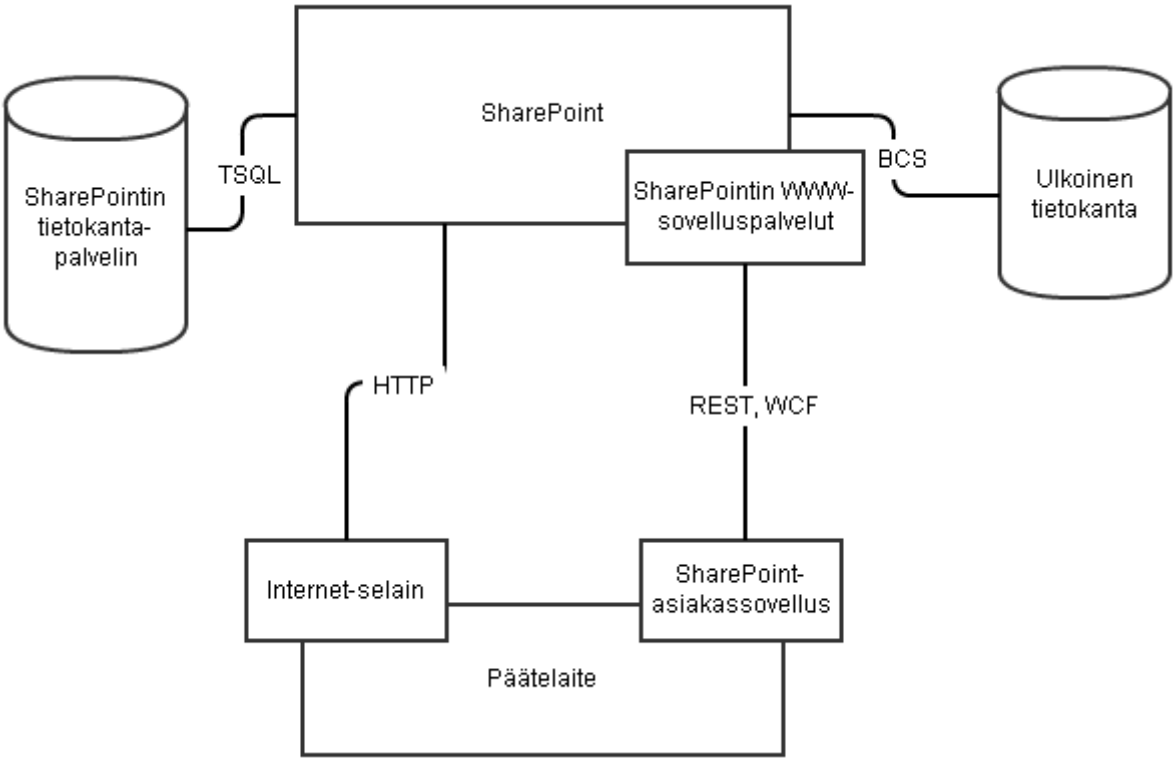
autentikointimalli, joka voidaan integroida käyttöorganisaatioissa laajemmin käytettäviin autentikointijärjestelmiin, voi säästää resursseja itse ohjelmiston toteutuksesta kun sitä ei tarvitse toteuttaa erikseen. [56]

Edellä mainittujen SharePointin abstraktioiden ja olemassa olevien toiminnallisuuksien lisäksi yksi SharePointin olennainen ominaisuus on sandbox-sovellukset. SharePoint-palvelimella suoritettavat ohjelmat voidaan toteuttaa joko niin, että niillä on suoritusympäristössään täydet oikeudet, tai niin, että ne suoritetaan rajatuilla oikeuksilla. Näitä rajattujen oikeuksien SharePoint-sovelluksia kutsutaan sandbox-sovelluksiksi, ja niillä on käytössään vain alijoukko SharePoint-luokkakirjastojen luokista. Sandbox-sovelluksien etuna on se, että niitä voidaan asentaa SharePoint-palvelimelle ilman järjestelmänvalvojien oikeuksia. Niiden käyttäminen on myös turvallista, sillä niiden käyttämiä järjestelmäresursseja voidaan rajoittaa palvelimen ylläpitäjien toimesta. Vastaavia täysien oikeuksien sovelluksia voivat asentaa vain järjestelmänvalvojat, ja niiden käyttämiä järjestelmäresursseja ei ole mahdollista rajoittaa. [56] Tästä johtuen sandbox-sovellukset ovat hyödyllinen ominaisuus esimerkiksi organisaatioissa, joiden tietosuojakäytännöt estävät täysien oikeuksien sovellusten käytön.

Edellä mainituista asioista voidaan todeta, että SharePoint tarjoaa ohjelmistonkehittäjille paljon vaihtoehtoja sekä erilaisia ominaisuuksia. SharePoint-ohjelmistokehityksessä käytettäviä teknologioita on useita, ja SharePointin omia, hyödynnettävissä olevia toimintoja on myös olemassa. Toisaalta, tässä voidaan nähdä myös SharePoint-ohjelmistokehityksen eräs heikkous: SharePoint pyrkii tarjoamaan jokaiselle ohjelmoijalle jotakin, mutta ohjelmistokehityksen näkökulmasta se ei erikoistu mihinkään tiettyyn osa-alueeseen. Samalla SharePointin sisältämät ominaisuudet tekevät SharePointista raskaamman alustan kuin mitä jotkut muut, riisutummat alustat ovat. Näin ollen pohdittaessa SharePointin käyttöä ohjelmistoprojekteissa tulisikin pohtia muun muassa sitä, että ovatko SharePointin tarjoamat ominaisuudet tarpeeksi spesifejä kyseiseen projektiin, ja että kuinka tärkeitä alustan tarjoamat ominaisuudet ovat verrattuna suorituskykyyn.

Yhteenvetona, SharePoint-ohjelmistokehityksessä voidaan luoda ohjelmia jotka suoritetaan

joko SharePoint-palvelimella tai käyttäjän päätelaitteella. Molemmissa tapauksissa ohjelmat käyttävät SharePointia tietovarastona, ja mahdollisesti hyödyntävät SharePointin BCS-ominaisuutta käyttääkseen ulkoisia tietokantoja. Käyttäjät käyttävät SharePoint-palvelimella suoritettavia ohjelmia Internet-selaimella, ja päätelaitteella ajettavia ohjelmia voidaan suorittaa joko Internet-selaimessa tai omassa prosessissaan. Päätelaitteella ajettavat ohjelmat kommunikoivat SharePoint-palvelimen kanssa WWW-sovelluspalveluiden välityksellä. Kuvassa 3 on esitettynä yksinkertaistettu SharePoint-arkkitehtuuri, jossa on kuvattuna SharePoint palvelin tietokantoineen sekä asiakassovellukset.



Kuva 3. Yksinkertaistettu SharePoint-arkkitehtuuri

3.2 Silverlight

Silverlight on Microsoftin kehittämä monipuolisten Internet-sovellusten teknologia, jota voidaan käyttää useissa eri selaimissa, käyttöjärjestelmissä ja päätelaitteissa. Silverlightin ensimmäinen versio julkaistiin vuonna 2007, ja tämän kirjoittamisen hetkellä viimeisin versio julkaistiin vuonna 2011. [65] Sen avulla voidaan kehittää visuaalisia ja monipuolisia

sovelluksia, jotka voivat sisältää multimediaelementtejä kuten kuvia, ääntä ja videoita. Silverlight-sovelluksia voidaan suorittaa joko Internet-selaimessa Silverlight-laajennuksen avulla tai omana erillisenä sovelluksenaan tietokoneen työpöydällä. Näin ollen vaikka Silverlight onkin pääasiassa monipuolisten Internet-sovellusten teknologia, ei sen käyttö rajoitu pelkästään tuohon tarkoitukseen. [52]

Teknologiana Silverlight pohjautuu Microsoftin .NET-ohjelmistokehykseen, ja Silverlight-ohjelmistokehitystä tehdäänkin .NET-ohjelmoinnilla. Yleisimpänä Silverlight-ohjelmoinnissa käytettävänä ohjelmointikielenä on .NETin C#. Käyttöliittymien toteuttamiseen Silverlightissa voidaan käyttää joko funktionaalista tai deklaratiivista ohjelmointia. Deklaratiivisesti ohjelmoitaessa Silverlight-sovelluksia käytetään XML:ään pohjautuvaa XAML-kieltä (Extensible Application Markup Language). XAML-ohjelmoinnissa määritellään käyttöliittymässä esiintyvät elementit sekä näiden ominaisuudet ja toiminnallisuudet staattisesti, kun taas käytettäessä C#-ohjelmointia käyttöliittymän elementtejä voidaan muokata ja lisätä vapaasti sovelluksen ajon aikana. Lisäksi C#-ohjelmoinnilla voidaan toteuttaa monimutkaisempaa ohjelmalogiikkaa, kun taas deklaratiivisella ohjelmoinnilla toteutettava toiminnallisuus rajoittuu käyttöliittymän manipuloimiseen. Silverlight-sovelluksia toteutettaessa ohjelmoijilla on käytettävinään myös .NETin luokkakirjastoja, joten .NETiin tutustuneille ohjelmistokehittäjille siirtyminen Silverlightiin on verrattain helppoa. [52]

Silverlight-sovellusten kehittämiseen on olemassa erilaisia työkaluja, kuten Microsoftin Visual Studio-ohjelmistot ja niin ikään Microsoftin kehittämä Expression Blend-ohjelmisto. Molemmilla näistä voidaan luoda täysin valmiita Silverlight-sovelluksia, mutta niiden tarjoamat työkalut soveltuvat hieman erilaisiin tarkoituksiin. Visual Studio on suunniteltu varsinaiseen ohjelmistokehitykseen, jossa pääpaino on funktionaalisella ohjelmoinnilla, vaikka siinä voidaan käyttää myös deklaratiivistakin ohjelmointia. Expression Blend sisältää laajemmat työkalut visuaaliseen suunnitteluun ja käyttöliittymien toteutukseen, ja se onkin suunnattu pääasiassa graafisille suunnittelijoille. Vaikka Expression Blendilläkin voidaan toteuttaa sovellukseen toiminnallisuutta, eivät sen ohjelmointityökalut kuitenkaan vastaa Visual Studion työkaluja. Työn alla oleva Silverlight-sovellus voidaan myös siirtää Visual Studiosta Expression Blendiin ja takaisin,

jolloin molempien ohjelmistojen tarjoamia ominaisuuksia voidaan hyödyntää yhdessä projektissa. [52]

Internet- ja työpöytäsovellusten lisäksi Silverlightilla voidaan toteuttaa myös sovelluksia mobiilialustoille. Esimerkiksi Windows Phone 7 –käyttöjärjestelmällä suoritettavat sovellukset tehdään pääasiassa Silverlightilla. Kuitenkin tällaisia sovelluksia tehtäessä Silverlight-sovelluskehityksessä on käytettävissä vain alijoukko Silverlightin ja .NETin luokkakirjastoista. Mutta huolimatta siitä, että mobiilisovelluksissa on käytettävissä vain rajattu osa Silverlightin toiminnallisuudesta, on tietokoneelle toteutettujen Silverlight-sovellusten kääntäminen myös mobiilialustalle suhteellisen helppoa. [52]

Monipuolisten Internet-sovellusten teknologioiden joukossa Silverlight on eräänlaisessa erityisasemassa tehtäessä SharePoint-sovelluksia. Tämä siksi, että SharePointissa on toteutettuna oma asiakassovellusten luokkakirjasto Silverlightille, joka helpottaa Silverlight-sovellusten integroimista SharePointiin. Tämä SharePointin luokkakirjasto hyödyntää SharePointin web-sovelluspalveluita, joten sen tarjoamat toiminnallisuudet ovat kyllä käytettävissä myös muillakin monipuolisten Internet-sovellusten teknologioilla kuin pelkästään Silverlightilla. Käytännössä valmiin luokkakirjaston käyttäminen on kuitenkin helpompaa kuin web-sovelluspalveluiden käyttäminen. Tästä johtuen Silverlightia voidaan pitää ensisijaisena monipuolisten Internet-sovellusten teknologiana toteutettaessa SharePoint-sovelluksia. [56]

4 TYÖSSÄ KÄYTETYT MENETELMÄT JA TYÖTAVAT

4.1 Asiakaskysely

Osana tätä työtä tehtiin Internetissä jaettava kyselytutkimus Onsight Helsinki Oy:n yhteistyökumppaneille, jossa selvitettiin SharePointia työkseen käyttävien henkilöiden mielikuvia SharePointista, SharePointin käytettävyydestä sekä käytettävyyden merkityksestä SharePoint-sovelluksissa. Kysely lähetettiin suoraan vastaajille sähköpostilla, ja saadun vastaukset käsiteltiin anonyymisti. Kyselytutkimuksen tuloksia analysoidaan sekä yksittäin että yhdessä, ja niiden pohjalta pyritään selvittämään kuinka merkittävää käytettävyyteen panostaminen oikeasti on erilaisissa ohjelmistoprojekteissa, erityisesti toteutettaessa SharePoint-sovelluksia. Analysoimalla kyselyn monivalintakysymyksiin saatuja vastauksia pyritään löytämään näiden kysymysten kohdalla yleisimpiä mielipiteitä. Kyselyn vapaiden vastausten avulla pyritään löytämään yksityiskohtaisia, mahdollisesti käytettävyyteen liittyviä asenteita, mielipiteitä ja ongelmia. Näiden analyysien pohjalta pyritään tekemään johtopäätöksiä käytettävyyden kehittämisestä ja sen merkityksestä SharePoint-sovelluskehityksessä asiakkaan näkökulmasta.

4.2 Työjono-ohjelman suunnittelu ja toteutus

Tämä työ pohjautuu SharePoint-alustalle toteutettuun Työjono-ohjelmistoon, joka toteutettiin Onsight Helsinki Oy:ssä ketteränä ohjelmistoprojektina. Työjonon suunnittelussa ja toteutuksessa edettiin tarpeen vaatimilla tavoilla siten, että työn tulokset annettiin ohjelmiston loppukäyttäjille validoitavaksi ja testattavaksi. Tämän validoinnin ja testauksen tulosten sekä muun loppukäyttäjiltä saadun palautteen pohjalta Työjonoon tehtiin vaadittuja muutoksia. Työjonon toteutuksesta vastaamassa oli yksi ohjelmistokehittäjä. Työjono-ohjelmistoon toteutettiin kaksi käyttöliittymää, joita käsitellään konstruktiivisen tutkimuksen keinoin. Näitä käyttöliittymiä vertaillaan keskenään etsimällä eroavaisuuksia niiden käytettävyydestä. Käytettävyysominaisuuksien välisiä eroja analysoidaan, ja analyysin tuloksista tehdään johtopäätöksiä siitä, miten SharePoint-sovellusten käytettävyyttä voidan parantaa Työjono-ohjelmiston toteutuksesta saatujen kokemusten valossa.

5 ASIAKASKYSELY SHAREPOINTISTA JA KÄYTETTÄVYYDESTÄ

Tässä työssä tehtiin Internetissä jaettava ja anonyymisti täytettävä kyselytutkimus SharePointista, siihen liittyvistä mielikuvista ja käytettävyyden merkityksestä SharePointin käytössä. Kyselyssä selvitettiin myös vastaajien käsityksiä SharePointin käytettävyyden laadusta. Kysely lähetettiin Onsight Helsinki Oy:n yhteistyötahoille ja asiakasyrityksille joissa SharePointia käytetään työntekoon. Sen tavoitteena oli selvittää mitä SharePoint-projekteihin liittyviä asioita suomalaiset SharePoint-toteutuksia tilaavat organisaatiot näkevät ongelmallisina, ja että kuinka tärkeäksi osaksi SharePoint-toteutuksia käytettävyys koetaan.

Kyselyn vastaukset kerättiin anonyymisti, ja se oli auki vastauksille kuuden viikon ajan keväällä 2012. Tänä aikana vastauksia saatiin seitsemän kappaletta, joten kyselyn otos jäi erittäin pieneksi. Kattavia yleistyksiä näin pienellä otoksella ei voida tehdä, mutta kyselyyn saaduista vastauksista ilmenee kuitenkin tiettyjä SharePoint-projekteissa huomioon otettavia asioita. Myös käytettävyyteen liittyvissä kysymyksissä saadut vastaukset olivat melko yksimielisiä, joten niitä voidaan pitää jossain määrin pätevinä. Taulukossa 2 on kaikki kyselyssä esitetyt kysymykset vastaustyyppeineen. Kaikki kyselyyn saadut vastaukset on kokonaisuudessa esitetty liitteessä 1.

Taulukko 1. Asiakaskyselyssä esitetyt kysymykset

Kysymys	Vastaustyyppi
”Mikä seuraavista vastaa sinua parhaiten?”	Valinta vaihtoehdoista ”Olen SharePoint-loppukäyttäjä”, ”Olen SharePoint-ylläpitäjä”, ”Olen SharePoint-kehittäjä” ja ”Ei mikään esitetyistä”.
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Vapaa kirjallinen vastaus.
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Valinta vaihtoehdoista ”intranet”, ”extranet” ja ”internet”.

”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Vapaa kirjallinen vastaus
”Miten SharePointin käyttö on vastannut odotuksianne?”	Vapaa kirjallinen vastaus
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	Valinta yhdestä viiteen. Viisi tarkoittaa erittäin tärkeää, yksi vähän tärkeää.
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	Valinta yhdestä viiteen. Viisi tarkoittaa erittäin tärkeää, yksi vähän tärkeää.
”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	Valinta yhdestä viiteen. Viisi tarkoittaa erittäin tyytyväinen, yksi ei-tyytyväinen.
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	Valinta yhdestä viiteen. Viisi tarkoittaa erittäin tärkeää, yksi vähän tärkeää.
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Vapaa kirjallinen vastaus
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Valinta kyllä / ei.

Kuten voitiin ehkä odottaakin, pitivät kaikki kyselyyn vastanneet käytettävyyttä merkittävänä ominaisuutena niin ohjelmistoissa ylipäätänsä kuin SharePoint-sovelluksissakin. Kuusi seitsemästä vastanneesta piti käytettävyyttä erittäin merkittävänä ja yksi seitsemästä piti sitä merkittävänä ominaisuutena. Toisaalta, käytettävyyden merkitystä selvittäneet kysymykset eivät asettaneet käytettävyyttä vastakkain minkään muun ominaisuuden kanssa, joiden väliltä vastanneiden olisi pitänyt valita merkittävämpi vaihtoehto. Näin ollen kysymyksiin vastaaminen havaitun laisesti oli helpompaa. Esimerkiksi, mikäli kysymyksissä olisi tiedusteltu kumpaa ominaisuutta pidetään merkittävämpänä, ohjelmiston käytettävyyttä vai luotettavuutta, olisivat vastaukset voineet olla hyvinkin toisenlaiset. Mutta koska tässä työssä ei ole tavoitteena asettaa käytettävyyttä vastakkain muiden ohjelmistoihin liitettyjen ominaisuuksien kanssa, vaan käsitellä sitä sellaisenaan, päädyttiin kyselyssä esittämään edellä mainittuja kysymyksiä.

Vastanneista viisi seitsemästä oli vain jokseenkin tyytyväisiä SharePointin käytettävyyteen. Tämän tuloksen pohjalta voidaankin todeta, että käytettävyyden kehittämiselle SharePointin osalta on tarvetta. On kuitenkin huomattava, että kyselyssä

selvitettiin nimenomaan vastanneiden kokemuksia SharePointin, eikä SharePoint-sovellusten käytettävyydestä. Näin ollen saadut vastaukset eivät kerro kolmannen osapuolen SharePoint-toteutusten käytettävyyden laadusta. Siitä huolimatta näiden tuloksien voidaan ajatella paljastavan erään markkinaraon SharePoint-sovelluksille. Nimittäin, koska SharePointin käytettävyydessä koetaan olevan parantamisen varaa, voisivat sellaiset kolmannen osapuolen SharePoint-sovellukset. jotka tekevät SharePointista käytettävämmän, olla kilpailukykyisiä.

Yleisesti ottaen kyselyyn vastanneiden odotukset SharePointia kohtaan olivat positiivisia. Käytännön kokemukset SharePointin käytöstä ovat kuitenkin olleet ristiriitaisia. Osa vastanneista oli tyytyväisiä tuotteeseen, vaikka he totesivatkin, että SharePointin loppukäyttäjien kiinnostus tai osaaminen SharePointin osalta on aiheuttanut ongelmia sen käyttöönotossa. Muille vastanneille SharePoint on aiheuttanut erilaisia ongelmia, mukaan lukien tarpeita ylimääräisille ohjelmistohankkeille niissä tapauksissa, joissa SharePointin omat ominaisuudet eivät ole olleet organisaation tarpeisiin riittäviä. Lähes kaikkien vastanneiden organisaatioissa on työasemissa käytettävissä Silverlight-laajennus, joten sen hyödyntäminen SharePoint-sovelluskehityksessä käytettävyyden parantamiseksi olisi näiden tulosten valossa usein mahdollista.

Kuten todettua, kyselyn otos oli erittäin pieni, eikä saatujen vastausten pohjalta voida näin ollen tehdä yleistäviä johtopäätöksiä. Tästä huolimatta näistä vastauksista voidaan nähdä, että käytettävyyttä pidetään tärkeänä ominaisuutena, ja että sen kehittämiselle SharePoint-sovelluksissa on tarvetta. Vapaisiin kysymyksiin saaduista vastauksista voidaan myös nähdä asioita, joita SharePointia käyttävissä organisaatioissa koetaan SharePointin kohdalla ongelmallisiksi. Koska näitä saatuja tuloksia ei voida yleistää, on käytettävyyteen liittyvälle, kattavammalle mielipidetutkimukselle yhä tarvetta. Silti näidenkin tulosten valossa voidaan todeta, että käytettävyyden kehittämiseen tähtäävälle työlle on tarvetta.

6 CASE: TYÖJONO

Tämä työ pohjautuu Onsight Helsinki Oy:ssä suunniteltuun ja toteutettuun SharePoint-alustalla suoritettavaan Työjono-ohjelmistoon, joka tehtiin sisäisesti käytettäväksi työajan seurantaan ja projektien hallintaan. Tässä luvussa esitellään tämä kyseinen Työjono-ohjelmisto sekä sen toteutukseen liittyvät vaiheet niiltä osin jotka liittyvät tämän työn aihepiiriin. Työjonolle toteutettiin kaksi erilaista käyttöliittymää. Projektin alkuvaiheessa toteutettiin ensin ASP.NET-pohjainen ja HTML:ään perustuva käyttöliittymä. Kuitenkin ensimmäisten Työjonon käytettävyystestausten jälkeen huomattiin, että tämä käyttöliittymä ei pystyisi tarjoamaan kaikkia niitä ominaisuuksia, joita Työjonon käyttöliittymältä toivottiin. Täten Työjonolle päätettiin toteuttaa erillinen monipuolisten Internet-sovellusten teknologiaan perustuva käyttöliittymä, joka oletettavasti tulisi tarjoamaan parempia käytettävyysominaisuuksia. RIA-teknologiaksi Työjonon kohdalla valittiin Silverlight, johtuen SharePointin Silverlightille tarjoamista integraatio-ominaisuuksista. Seuraavassa esitellään tarkemmin Työjonon suunnittelua, jonka jälkeen käsitellään Työjonon toteutusta. Lopuksi vertaillaan Työjonon ASP.NET- ja Silverlight-käyttöliittymien eroja käytettävyyden näkökulmasta.

6.1 Työjonon suunnittelu

Työjono-ohjelmiston tarkoituksena on toimia avustavana työkaluna työajan seurannassa sekä projektinhallinnassa. Tätä varten Työjono tulisi sisältämään kaksi keskeistä toimintoa. Ensimmäiseksi, Työjono tarjoaisi yrityksen projektipäälliköille keskitetyn näkymän projektien työntekijöiden ajankäyttöön ja näiden tekemiin projektipäivityksiin. Projektipäivitykset ovat työntekijöiden luomia viestejä, joissa kerrotaan muutoksista projektin ja siihen kuuluvien töiden tilassa, ja ne voivat myös tarvittaessa sisältää liitetiedostoja. Projektipäällikön olisi myös mahdollista nähdä työntekijöiden kalentereihinsa tekemät ajanvaraukset yhdestä näkymästä, jolloin uusien ajanvarausten tekeminen olisi helpompaa. Samassa näkymässä näkyisi myös projektipäällikön Työjono-ohjelmiston kautta työntekijöille luomat ajanvaraukset. Toiseksi, Työjonon avulla projektin työntekijöiden olisi mahdollista tehdä projektipäivityksiä keskitetysti yhteen paikkaan. Ennen Työjonoa muutoksista projekteissa viestitettiin useilla eri kanavilla, kuten

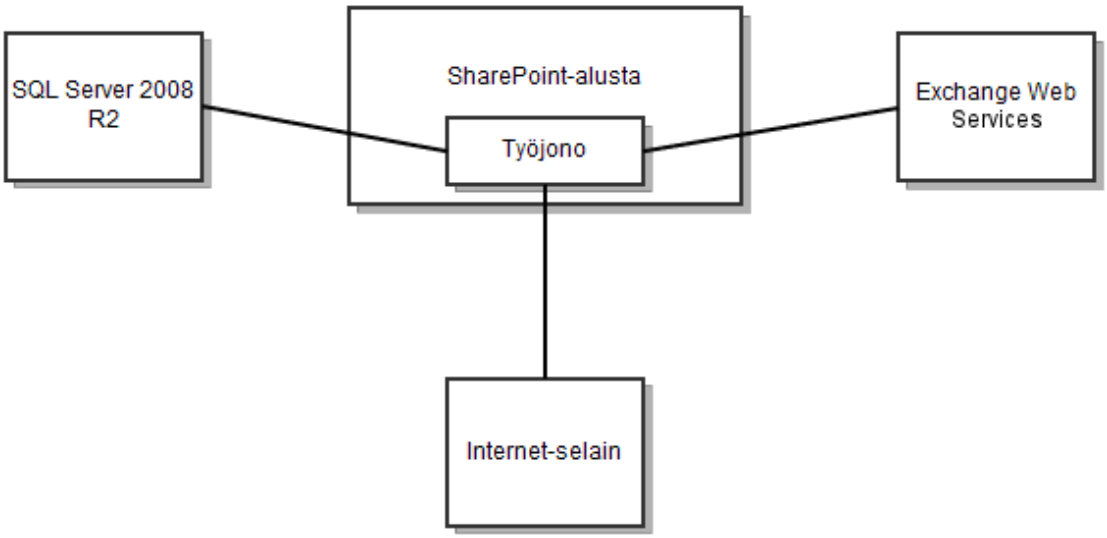
sähköpostitse ja pikaviestimillä, joten keskitetylle kommunikaatiokanavalle oli tarvetta.

Työjono toteutettiin ketteränä projektina, jossa ohjelmiston loppukäyttäjänä toimiva projektinhallinnan henkilö oli koko projektin ajan mukana suunnittelussa ja testauksessa. Näin ollen ohjelmiston käyttöliittymäsuunnittelussa voitiin ottaa huomioon myös käyttäjien näkökulma ja ohjelmistoon pystyttiin tarvittaessa tekemään loppukäyttäjien toivomia muutoksia lyhyelläkin aikavälillä. Toisaalta, johtuen Työjono-projektin ketterästä luonteesta ei projektissa tehty etukäteen suuria ja tarkkoja suunnitelmia siitä, millainen ohjelmistosta pitäisi nimenomaan tulla, tai miten se pitäisi toteuttaa.

Työjono päätettiin toteuttaa SharePoint-alustalla, koska se tarjoaa valmiina muun muassa autentikointiominaisuuksia, jolloin näitä ei tarvitsisi toteuttaa erikseen osana Työjono-ohjelmistoa. Myös SharePointin abstraktioiden, kuten tietokantojen tauluja vastaavien luetteloiden, koettiin helpottavan ja nopeuttavan kehitystyötä. Lisäksi nämä abstraktiot tulisivat myös helpottamaan ohjelmiston ylläpitoa, kun niitä voitaisiin manuaalisesti käsitellä SharePointin omien käyttöliittymien kautta ilman erityisosaamista tai erillisiä työkaluja [34]. Näin ollen Työjonon toteutus voitaisiin tehdä nopeammin kun toteutettavia ohjelmiston osia olisi vähemmän, ja täten ensimmäiset testiversiot saataisiin nopeammin testattavaksi.

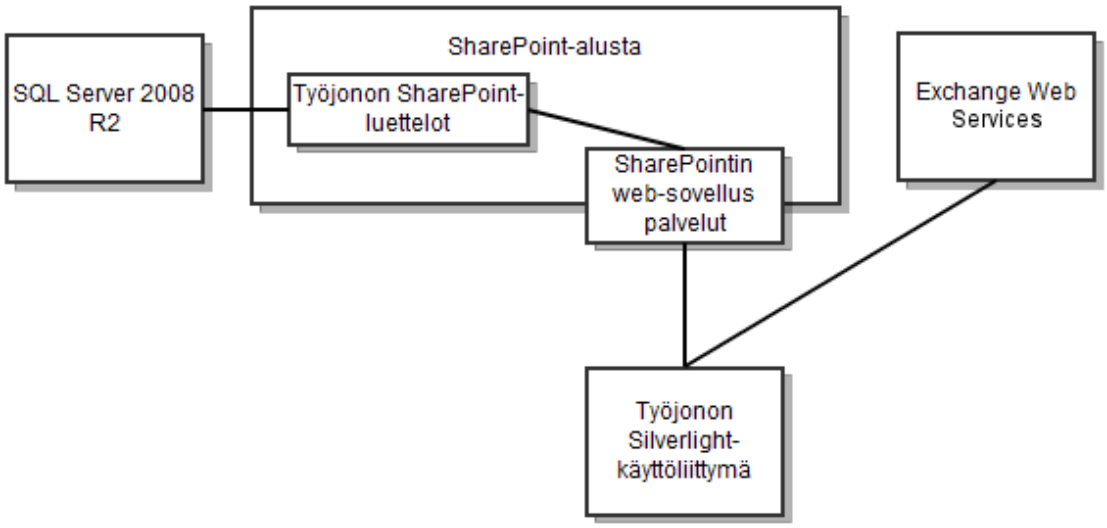
Ensimmäinen Työjonolta vaadittavista toiminnoista oli keskitetty näkymä, josta projektipäällikön olisi mahdollista seurata projektien etenemistä sekä työntekijöiden ajankäyttöä. Tämä tulisi tapahtumaan kahdella tavalla: Työjono näyttäisi yhdessä näkymässä sekä työntekijöiden luomat projektipäivitykset, työntekijöille määrätyt projektitehtävät että työntekijöiden sähköiset kalenterit ajanvarauksineen. Koska Onsight Helsinki Oy:n työntekijät käyttävät Microsoft Outlook-kalentereita, tulisi Työjonon myös olla integroitu Microsoftin Exchange Serveriin. Exchange Server toteuttaa Outlook-kalentereiden toiminnallisuuden, joita voidaan ohjelmallisesti hyödyntää Exchange Web Service-sovelluspalveluiden (EWS) avulla [66]. Tässä vaiheessa Työjono-ohjelmiston arkkitehtuurikuvaukseen kuuluvat seuraavat osat: SharePoint-alusta, SharePoint-alustalla suoritettava Työjono-ohjelmisto, EWS, Microsoft SQL Server 2008 R2-tietokanta ja loppukäyttäjän käyttämä Internet-selain. Kuvassa 4 on esitetty Työjonon

arkkitehtuurikaavio tässä vaiheessa.



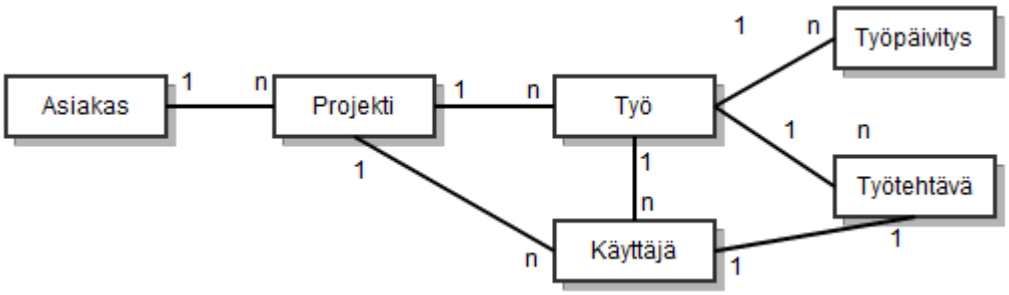
Kuva 4. Työjono-ohjelmiston ensimmäisen vaiheen arkkitehtuurikaavio

Kun Työjonosta oli toteutettu ensimmäinen testiversio todettiin, että ASP.NET-pohjainen käyttöliittymä ei tulisi tarjoamaan kaikkia Työjonolta vaadittavia käytettävyysominaisuuksia. Erityisesti työajanseurantaan ja projektien hallinnointiin käytettävää yhtenäistä näkymää ei voitaisi toteuttaa niin kuin olisi tarkoitus pelkän HTML:n avulla. Tämän takia Työjonoon päädyttiin toteuttamaan myös Silverlight-käyttöliittymä. Tämän muutoksen myötä Työjonon uudessa arkkitehtuurikaaviossa (kuva 5) ovat seuraavat osat: SharePoint-alusta, tällä olevat Työjonon SharePoint-luettelot, SharePointin web-sovelluspalvelut, EWS, Microsoft SQL Server 2008 R2-tietokanta ja loppukäyttäjän päätelaitteessa suoritettava Työjonon Silverlight-käyttöliittymä.



Kuva 5. Työjonon toisen vaiheen arkkitehtuurikaavio

Työjono-ohjelmisto sisältää kuusi tietotyyppiä: käyttäjä, asiakas, projekti, projektin työ, työn työtehtävä sekä työn päivitys. Käyttäjät kuvaavat Työjonon varsinaisia käyttäjiä, ja niiden avulla hallitaan ohjelmiston käyttäjärooleja. Asiakkaat vastaavat yrityksen asiakkaita, ja niillä voi olla n kappaletta projekteja. Projektit vastaavat asiakkaille toteutettavia projekteja, ja kullakin projektilla voi olla n kappaletta töitä. Projekteihin liitetään myös niihin kuuluvat käyttäjät. Projektien työt vastaavat pienempiä töitä jotka yhdessä muodostavat projektin, kuten esimerkiksi ”suunnittelu” ja ”toteutus.” Myös töihin liitetään niihin kuuluvat käyttäjät. Töillä voi olla n kappaletta työtehtäviä, jotka ovat projektipäällikön asettamia tiettynä aikana ja tietyn henkilön suoritettavia tehtäviä. Työtehtävät ovat toteutettuna vain Työjonon Silverlight-käyttöliittymässä. Työpäivitykset, tai projektipäivitykset, ovat tietylle työlle osoitettuja käyttäjien lisäämiä päivityksiä. Työtehtäviin ja –päivityksiin voidaan myös liittää liitetiedostoja. Kuvassa 6 on mallinnettuna Työjonon tietotyypit ja näiden väliset suhteet.



Kuva 6. Työjonon tietotyypit ja niiden väliset suhteet.

Riippumatta käytettävästä käyttöliittymästä Työjonon käyttäjilleen tarjoamien toimintojen tulisivat olla samat, vain toimintojen tekotapa sekä asioiden esitysmuoto vaihtuisi. Poikkeuksena tähän on Silverlight-käyttöliittymässä toteutetut työtehtävät, jotka päätettiin toteuttaa vasta sen jälkeen, kun ASP.NET-käyttöliittymän kehitystyö oltiin lopetettu. Työjonon käyttöliittymän kautta tehtävät toiminnot voidaan jakaa kahteen ryhmään käyttäjän roolin mukaan: Työntekijän toimintoihin sekä projektipäällikön toimintoihin. Ylläpitotoiminnot, kuten käyttäjien hallinta, voidaan tehdä SharePointin tarjoamien luetteloiden hallintanäkymien kautta, joten näille ei Työjonossa toteutettu erillistä käyttöliittymää. Taulukossa 2 on listattuna Työjonon käyttöliittymän kautta tehtävät toiminnot käyttäjän roolin mukaan.

Taulukko 2. Työjonon käyttöliittymän tarjoamat toiminnot

Työntekijän toiminnot	**Projektipäällikön toiminnot**
Luo päivityksiä omiin työtehtäviin	Luo päivityksiä kaikkiin projekteihin
Tarkastele omien projektien ja töiden edistymistä	Tarkastele kaikkien projektien ja töiden etenemistä
Tarkastele omia ajanvarauksia Outlook-kalenterista	Tarkastele kaikkien työntekijöiden ajanvarauksia Outlook-kalentereista
Tarkastele omia ajanvarauksia työjono-ohjelmasta	Tarkastele kaikkien työntekijöiden ajanvarauksia työjono-ohjelmassa
	Luo uusia tehtäviä, töitä, projekteja ja asiakkaita Työjonoon

6.2 Työjonon toteutus

Työjono-ohjelmiston toteutuksesta voidaan erottaa kolme erillistä vaihetta: Työjonon tietovarastoina toimivien SharePoint-luetteloiden toteutus, Työjonon ASP.NET-käyttöliittymän toteutus sekä Työjonon Silverlight-käyttöliittymän toteutus. Koska Työjonon tietovarastojen toteutus ei ole tämän työn osalta oleellista, ei sitä käsitellä tarkemmin. Sen sijaan seuraavassa esitellään sekä ASP.NET- että Silverlight-käyttöliittymien toteutukset, kyseiset käyttöliittymät sekä käyttöliittymien ominaispiirteitä ja toimintalogiikkaa.

6.2.1 Työjonon ASP.NET-käyttöliittymä

Työjonon ASP.NET-käyttöliittymä toteutettiin web-osina (eng. *web part*), jotka ovat modulaarisia ASP.NET-sivuille asetettavia ohjelmallisia komponentteja [56]. Työjonossa on yhteensä neljä erillistä web-osaa: Työntekijän seurannan web-osa, projektinseurannan web-osa, työn luonnin web-osa sekä projektipäivityksen luonnin web-osa. Triviaaleimmat projektipäällikön toiminnot, kuten asiakkaiden ja projektien lisääminen Työjonoon, jätettiin tehtäväksi SharePointin luettelonäkymillä, koska niiden toteuttaminen ASP.NETillä ei olisi tuonut nähtävää lisäarvoa.

ASP.NET-käyttöliittymistä projektinseurannan ja projektipäivitysten web-osat ovat kaikkien käyttäjien käytettävissä. Sen sijaan työntekijöiden seurannan sekä työtehtävien luonnin web-osat on tarkoitettu pelkästään projektipäällikön käytettäviksi. Projektipäivitysten ja työtehtävien luonnin web-osat ovat käytännössä yksinkertaisia HTML-lomakkeita, jotka täyttämällä ja lähettämällä käyttäjä tallentaa tietoa Työjonon SharePoint-luetteloihin. Kuvat Työjonon ASP.NET-käyttöliittymästä on liitteessä 2.

Projektin- sekä työntekijänseurannan web-osat ovat hieman monimutkaisempia kuin projektipäivitysten ja työtehtävien luonnin web-osat, ja suurimmat käytettävyysodotukset kohdistuvatkin nimenomaan niihin. Projektinseurannan web-osalla käyttäjän, joko projektin työntekijän tai projektipäällikön, tulisi pystyä seuraamaan projektin tilaa tarkistelemalla projektiin liitettyihin töihin luotuja projektipäivityksiä. Käytännössä tämä tapahtuu siten, että käyttäjä valitsee ensiksi asiakkaan jonka projekteja tarkastelee, toiseksi kyseisen asiakkaan projekteista halutun projektin ja kolmanneksi valitun projektin töistä halutun työn. Kun työ on valittu, tulee käyttäjän valita tarkastellaanko työtä vai muokataanko sitä. Tarkasteltaessa web-osa esittää työn tiedot ja päivitykset liitteineen.

Johtuen käytetystä teknologiasta, kaikki käyttäjän projektiseurannan web-osassa tekemät toiminnot luovat uuden HTTP-pyynnön SharePoint-palvelimelle ja toiminnon tulokset vastaanotetaan uutena HTML-sivuna. Näin ollen yhden työn tarkastelussa joudutaan tekemään kolme erillistä HTTP-pyyntöä: Ensimmäinen valittaessa asiakas, toinen valittaessa asiakkaan projekti ja kolmas avattaessa projektiin kuuluva työ. Tämä tekee

web-osan käyttämisestä hidasta, kun triviaalienkin toimintojen jälkeen HTML-sivu joudutaan lataamaan palvelimelta uudestaan.

Työntekijänseurannan web-osa toimii samankaltaisesti kuin projektinseurannan web-osa: Käyttäjä valitsee web-osassa olevista listoista ensiksi tarkasteltavan työntekijän. Tämän jälkeen web-osa näyttää työntekijän viimeisimmät projektipäivitykset sekä kalenterin, jossa on merkattu ne päivät joilla on kalenterivarauksia. Käyttäjä voi suodattaa näytettäviä projektipäivityksiä valitsemalla web-osassa olevista listoista jonkin tietyn projektin, jossa kyseinen työntekijä on mukana, ja mahdollisesti vielä kyseiseen projektiin liittyvän työn. Kun käyttäjä klikkaa kalenterista päivää jolla työntekijällä on kalenterivarauksia, niin web-osa näyttää myös tarkempia tietoja kyseisen päivän tapahtumista.

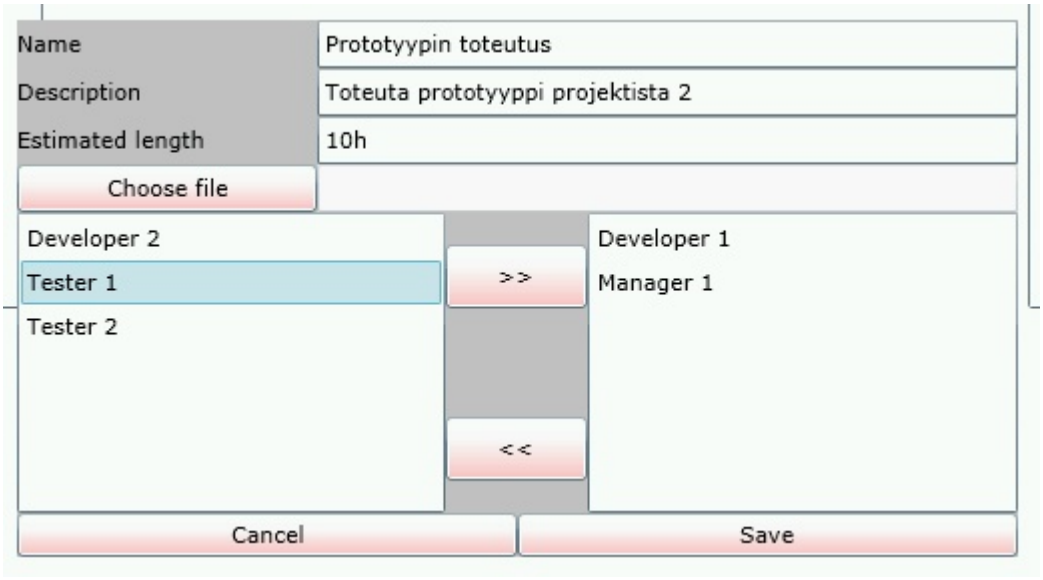
Kuten projektiseurannan web-osassa, myös työntekijänseurannan web-osassa kaikki tehtävät toiminnot vaativat HTTP-pyynnön lähettämistä palvelimelle ja täten uuden HTML-sivun lataamista. Johtuen siitä, että web-osissa käytetään HTML:ää tiedon esittämiseen ovat käytettävissä olevat esitystavat rajalliset. Tämän takia esimerkiksi kalenterin esittäminen muodossa, josta ajanvaraukset näkyisivät helposti, on vaikeaa ja sen sijaan käyttäjä joutuu valitsemaan tiettyjä päiviä tarkemman tarkastelun kohteeksi. Lopulta, koska HTTP ja ASP.NET ovat tilattomia teknologioita, on monimutkaisemman toiminnallisuuden toteuttaminen niiden avulla raskaampaa, koska kaikki tieto ohjelman nykyisestä tilasta tulee sisällyttää jokaiseen palvelimelle lähetettävään HTTP-pyyntöön. [64]

6.2.2 Työjonon Silverlight-käyttöliittymä

Työjonon Silverlight-käyttöliittymä toteutettiin yhtenä Silverlight-sovelluksena, joka sisältää neljä eri näkymää: Asiakasnäkymän, projektinäkymän, työnäkymän sekä kalenterinäkymän. Asiakasnäkymässä projektipäällikön roolissa oleva käyttäjä pystyy luomaan uusia asiakkaita Työjonoon sekä lisäämään asiakkaille uusia projekteja. Vastaavasti projektinäkymässä projektipäällikkö voi luoda projekteille uusia töitä tai tarkastella ja muokata olemassa olevia projekteja. Työnäkymässä käyttäjä voi luoda töille uusia työtehtäviä tai työpäivityksiä sekä tarkastella jo olemassa olevia työtehtäviä ja –päivityksiä valitun työn perusteella. Yhdessä nämä kolme näkymää vastaavat Työjonon

ASP.NET-käyttöliittymän projektiseurannan, työn luomisen ja työpäivityksen luomisen web-osia.

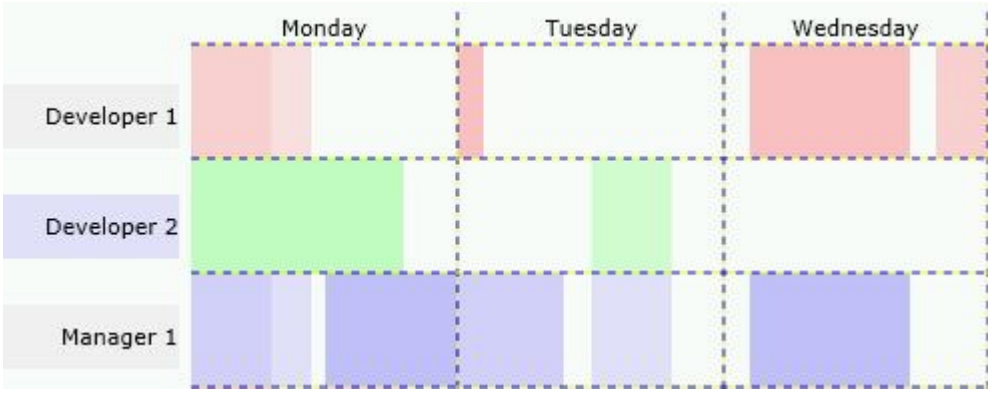
Kalenterinäkymässä, joka vastaa ASP.NET-käyttöliittymän työntekijäseurannan web-osaa, käyttäjä pystyy tarkastelemaan työntekijöiden sähköisiä kalentereita sekä näiden tekemiä projektipäivityksiä. Se koostuu kahdesta osasta: Näkymän yläreunassa on neljä listaa, yksi projekteille, yksi töille, yksi työtehtäville ja yksi työpäivityksille. Näissä listoissa näytetään tarkempaan tarkasteluun valitun työntekijän projektit, työt, työtehtävät sekä työpäivitykset. Listojen alla on viikkokalenteri, jossa kuvataan valittujen työntekijöiden ajanvaraukset valitulle kalenteriviikolle. Kalenterissa näytettävät työntekijät valitaan kalenterin oikealla puolella olevilla valintaruuduilla. Kaikki valitut työntekijät ja näiden ajanvaraukset näytetään kalenterissa allekkain niin, että käyttäjän on helppo nähdä milloin työntekijöillä on vaikkapa yhteistä varaamatonta työaikaa. Kalenteri näyttää päivittäiset ajanvaraukset kello kahdeksan aamulla ja kello kuusi iltapäivällä välillä, joten tämän välin ulkopuolella olevat ajanvaraukset jäävät näkymättä kalenterista. Kuvassa 7 on esimerkki työn luomisen lomakkeesta projektinäkymässä.



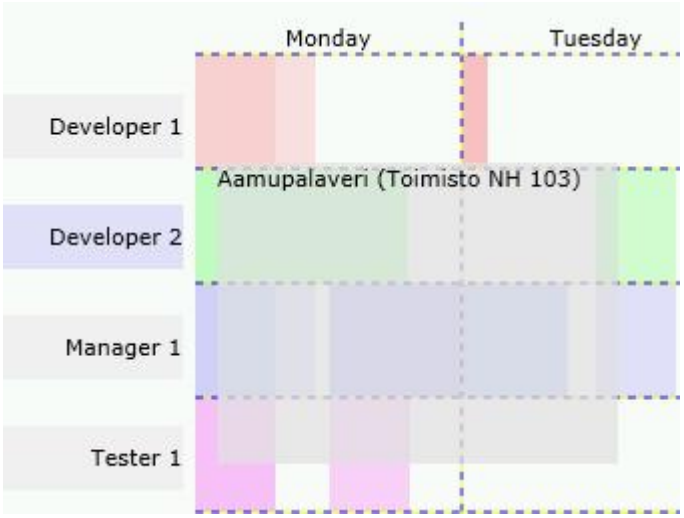
Kuva 7. Työn luominen Työjonon Silverlight-käyttöliittymän projektinäkymässä

Kalenterinäkymässä voidaan valita yksi työntekijä kerrallaan tarkempaan tarkasteluun. Tällöin kalenterinäkymässä oleviin projekti-, työ-, työtehtävä- ja työpäivitys-listoihin tulee esille kyisen työntekijän projektit, työt, työtehtävät sekä työpäivityksen. Käyttäjän on

mahdollista tarkastella näiden tietoja valitsemalla haluttu kohde listoista ja painamalla käyttöliittymässä olevaa ”Näytä kohde”-nappia. Mikäli käyttäjä valitsee listoista työtehtävän, korostetaan kyseisen työtehtävän ajanvaraus kalenterista mikäli se on näkyvissä. Jos taas käyttäjä valitsee työn, korostetaan kalenterista kaikki kyseiseen työhön liittyvän työtehtävät. Vastaavasti valittaessa projekti korostetaan kalenterista kaikkiin projektin töihin kuuluvat kaikki työtehtävät. Täten käyttäjän on helppo nähdä, miten kullekin projektille tai työlle on varattu aikaa. Lopuksi, viedessään hiiren osoittimen kalenterissa olevan ajanvarauksen päälle käyttöliittymä näyttää kyseiseen ajanvaraukseen liittyviä tietoja. Kuvassa 8 on esitettynä osa viikkokalenterista, jossa on valittuna useita työntekijöitä. Kuvassa 9 on viikkokalenteri, jossa yhdestä ajanvarauksesta on näkyvillä lisätietoja. Lisää kuvia Työjonon Silverlight-käyttöliittymästä on liitteessä 3.



Kuva 8. Otos Työjonon Silverlight-käyttöliittymän viikkokalenterista, jossa on tarkasteltavana kolme työntekijää.



Kuva 9. Otos Työjonon Silverlight- käyttöliittymän viikkokalenterista, jossa on

näkyvillä ajanvarauksen tarkempia tietoja

Työjonon Silverlight-käyttöliittymässä siirtyminen näkymien välillä on helppoa ja nopeaa verrattuna ASP.NET-käyttöliittymään, jossa eri web-osat voivat olla SharePoint-sivustolla eri sivuilla. Pääasiallinen tapa siirtymiseen näkymien välillä on käyttämällä käyttöliittymässä olevia navigointinappeja, jotka siirtävät käyttäjän suoraan valittuun näkymään. Tämän lisäksi asiakas- ja projektinäkymistä voidaan siirtyä hierarkiassa alaspäin: Asiakasnäkymästä projektinäkymään ja projektinäkymästä työnäkymään. Esimerkiksi, kun käyttäjä valitsee asiakasnäkymässä tietyn asiakkaan näkymään tulee esille kyseisen asiakkaan projektit, ja valittuaan näistä projekteista yhden käyttäjä siirtyy suoraan projektinäkymään. Kun projektinäkymään siirrytään näin, niin siinä näytettävät projektit suodatetaan asiakasnäkymässä valitun asiakkaan mukaan. Vastaavasti, kun projektinäkymässä valitaan jokin projekti, tulee näkyville kyseisen projektin työt. Jos käyttäjä valitsee näistä töistä jonkin, siirtyy hän suoraan työnäkymään, jossa tämä työ on valmiiksi valittuna ja muut valittavissa olevat työt on suodatettu projektin mukaan.

Toisin kuin ASP.NET-käyttöliittymässä, Silverlight-käyttöliittymässä iso osa toiminnoista voidaan tehdä nopeasti pelkästään käyttäjän päätelaitteella ilman kommunikaatiota SharePoint-palvelimen kanssa. Silverlight-käyttöliittymä lataa tärkeimpiä tietoja, kuten Työjonossa olevat asiakkaat, projektit ja työt, välimuistiin, jolloin näiden tarkastelu ja näkymien välillä siirtyminen voidaan tehdä nopeasti. Kuitenkin osa Silverlight-käyttöliittymän toiminnoista vaativat pyyntöjen tekemistä palvelimelle. Näitä toimintoja ovat uusien tietueiden, kuten asiakkaiden, projektien ja töiden, luominen sekä viikkokalenterin selaaminen, jossa Outlook-kalenterin ajanvarauksia ei voida etukäteen ladata välimuistiin.

6.3 Työjonon käytettävyys

Kuten edellä nähtiin, ovat Työjonon ASP.NET- ja Silverlight-käyttöliittymät keskenään hyvinkin erilaisia, ja käytettävyyden kannalta niiden välillä on useita eroja. Osa näistä eroista on kokonaan monipuolisten Internet-sovellusten teknologioiden mahdollistamia, kun taas jotkut ominaisuudet olisivat toteutettavissa molemmissa käyttöliittymässä tietyin rajoituksin. Seuraavaksi näitä käyttöliittymien välisiä käytettävyyseroja käsitellään eri

toimintojen mukaan, jotka on listattu taulukossa 2. Käyttöliittymien kuvankaappaukset, jotka liittyvät tässä luvussa käsiteltäviin vertailuihin, on nähtävissä liitteissä 2 ja 3.

Ensimmäisenä tarkasteltavana toimintona on projektipäivitysten luominen. Työjonon ASP.NET-käyttöliittymässä tämä tehdään yksinkertaisella lomakkeella, jossa käyttäjä valitsee työn johon päivitys liitetään, työn uuden tilan, esimerkiksi ”työn alla” tai ”valmis”, nimen päivitykselle sekä kuvauksen. Käyttäjä voi myös liittää päivitykseen liitetiedoston. Silverlight-käyttöliittymän työnäkymässä käyttäjä valitsee työn johon päivitys lisätään ja painaa ”Uusi päivitys”-nappia. Tämän jälkeen käyttäjälle näytetään lomake, johon täytetään päivityksen nimi ja kuvaus, valitaan työn uusi tila sekä liitetään mahdollinen liitetiedosto.

Kuten voitiin ehkä odottaa, ei projektipäivitysten tietojen syöttämisessä ole merkittäviä eroavaisuuksia käyttöliittymien välillä. Sen sijaan oikean työn valitsemisessa on eroja: ASP.NET-käyttöliittymässä käyttäjän tulee valita päivitykseen liittyvä työ kaikkien käyttäjälle kuuluvien töiden joukosta. Tämän sijasta Silverlight-käyttöliittymässä käyttäjä voi halutessaan joko suodattaa valittavat työt projektin mukaan tai valita työn kaikkien töiden joukosta. Samalla Silverlight-käyttöliittymä näyttää tietoja valitusta työstä, jolloin oikean työn löytäminen on helpompaa. Toisaalta nämä ominaisuudet, eli töiden suodattaminen projektin mukaan sekä töiden tietojen näyttäminen, olisi mahdollista toteuttaa myös ASP.NET-käyttöliittymässä. Kuitenkin tässä tapauksessa toteutus ei olisi yhtä käytettävä kuin Silverlight-käyttöliittymässä, sillä sekä töiden suodattaminen että kunkin työn tietojen näyttäminen tarvitsisi ainakin yhden HTTP-pyynnön lähettämisen lisää. Näin ollen ASP.NETillä toteutettuna nämä ominaisuudet olisivat selvästi hitaampia käyttää kuin Silverlightilla toteutettuna, jossa töiden tiedot voidaan ladata välimuistiin ja piirtää näkyville tarpeen mukaan.

Toisena tarkasteltavana toimintona on projektien ja töiden edistymisen seuraaminen. ASP.NET-käyttöliittymässä tämä tehdään projektinseurannan web-osalla, joka näyttää projektien edistymisen projekteihin liitettyjen töiden mukaan. Käyttäjä valitsee käyttöliittymässä asiakkaan, asiakkaalle kuuluvan projektin sekä projektiin kuuluvan työn. Valittuaan työn ja painettuaan ”Näytä työ”-nappia käyttäjälle näytetään kyseisen työn

tiedot sekä kaikki työhön liittyvät työpäivitykset, jotka kuvaavat työn edistymistä. Silverlight-käyttöliittymässä käyttäjä tekee samat asiakas-, projekti- ja työvalinnat. Valittuaan työn käyttäjälle näytetään tiedot työstä sekä listat työhön liittyvistä työtehtävistä ja –päivityksistä, joiden tietoja voidaan tarkastella valitsemalla ne listoilta. Työpäivitykset on järjestetty luontiajan mukaan siten, että uusin päivitys on aina listan ensimmäisenä päivityksenä.

Projektin ja työn seurannan edistymisessä ei ole toiminnallisuuteen liittyviä merkittäviä käytettävyyseroja: Sekä ASP.NET- että Silverlight-käyttöliittymässä käyttäjän tekemät toiminnot ovat periaatteessa samat. Merkittävin ero näiden toteutusten välillä on siinä, että ASP.NET-käyttöliittymässä käyttäjän valittua asiakkaan, projektin tai työn tehdään uusi HTTP-pyyntö palvelimelle, joka hidastaa web-osan käyttöä. Sen sijaan Silverlight-käyttöliittymässä näitä pyyntöjä ei tehdä, vaan käyttöliittymä pääsee siirtymään seuraavaan tilaan heti kun valinta on tehty. Toinen ero on työpäivityksien esitystavassa. ASP.NET-käyttöliittymässä kaikki työpäivitykset näytetään heti kun työ on valittu, kun taas Silverlight-käyttöliittymässä ne pitää avata yksitellen. Käytettävyyden kannalta se, että kumpi näistä tavoista on parempi, riippunee tilanteesta. Mikäli työssä on paljon työpäivityksiä, voi käytettävyys heikentyä jos ne kaikki esitetään sellaisenaan. Mutta jos päivityksiä on vähän, voisi käytettävyyden kannalta olla parempi että ne on helposti nähtävillä. Toisaalta, Silverlight-käyttöliittymässä on mukana myös työtehtävät, joten siinä esitettävää tietoa on joka tapauksessa enemmän kuin ASP.NET-käyttöliittymässä.

Kolmantena toimintona on Outlook-kalenterin ja, Silverlight-käyttöliittymän tapauksessa, Työjonon ajanvarausten tarkastelu. ASP.NET-käyttöliittymässä tämä tehdään työntekijän ajanseurannan web-osalla. Web-osassa käyttäjä valitsee tarkasteltavan työntekijän, jonka jälkeen käyttöliittymässä näkyy kyseisen työntekijän viimeisimmät projektipäivitykset sekä kalenteri. Tähän kalenteriin on merkattuna ne päivät, joille kyseisellä työntekijällä on ajanvarauksia. Käyttäjä voi suodattaa näytettävät projektipäivitykset valitsemalla käyttöliittymän listoilta tietyn projektin ja projektiin kuuluvan työn. Kalenterissa olevista ajanvarauksista saadaan tietoja valitsemalla ne hiirellä.

Silverlight-käyttöliittymässä työntekijän ajanseuranta tehdään hyvin erilaisella

kalenterinäkymällä. Kalenterinäkymässä käyttäjä voi valita yhdestä kuuteen työntekijää tarkasteltavaksi yhtä aikaa, ja kuukausikalenterin sijasta näytetään viikkokalenteri. Kalenterissa olevista ajanvarauksista saadaan tietoja suoraan viemällä hiiren osoitin varauksen päälle. Käyttäjä voi tarkastella yksittäisen työntekijän työpäivityksiä ja työtehtäviä klikkaamalla työntekijän nimeä, jonka jälkeen päivitykset ja tehtävät tulevat näkyviin listoihin. Valitsemalla listoissa olevia tehtäviä, töitä ja projekteja nämä korostuvat myös viikkokalenterissa.

Käytettävyyden kannalta näissä toteutuksissa on paljon eroavaisuuksia. Ensinnäkin ASP.NET-toteutuksessa käyttäjä pystyy tarkastelemaan vain yhtä työntekijää kerrallaan, kun taas Silverlight-käyttöliittymässä tarkasteltavia työntekijöitä voi olla yhdestä kuuteen. Tästä johtuen käyttö on tehokkaampaa, kun käyttäjä voi halutessaan saada esille enemmän tietoa kerrallaan. Lisäksi nämä tiedot esitetään muodossa, jossa tietojen väliset erot on helposti havaittavissa. Käytännössä jonkinlainen usean työntekijäin yhtä aikainen tarkastelu olisi myös toteutettavissa ASP.NET-käyttöliittymässä. Kuitenkin johtuen HTML:n rajoitetuista esitystavoista sekä useista HTTP-pyynnöistä, joita tällainen käyttöliittymä vaatisi, olisi tällaisen toteutuksen käytettävyys kuitenkin heikompaa kuin monipuolisten Internet-sovellusten teknologioita hyödyntäen. Toinen itse kalenteriin liittyvä eroavaisuus on siinä, että ASP.NET-käyttöliittymässä on kuukausikalenteri viikkokalenterin sijaan. Päätös tehdä Työjonon lopulliseen käyttöliittymään viikkokalenteri perustuu käyttäjiltä saatuun palautteeseen, ja se olisi toteutettavissa myös HTML:llä. Sen sijaan Silverlight-käyttöliittymässä toteutettua ajanvarausten tietolaatikkoa, joka tulee esille vietäessä hiiren osoitin ajanvarauksen päälle, ei voitaisi sellaisenaan tehdä pelkällä ASP.NETillä [64].

Kalenteriin liittyvien käytettävyyseroavaisuuksien lisäksi eroja on myös työntekijöiden tekemien työpäivitysten tarkastelussa. ASP.NET-käyttöliittymässä näytetään työntekijän viisi viimeisintä työpäivitystä suoraan siten, että kaikki niihin liittyvät tiedot ovat heti nähtävissä. Tämän sijaan Silverlight-käyttöliittymässä päivitykset tulee avata yksitellen. Käytettävyyden kannalta ASP.NET-käyttöliittymän voidaan ajatella olevan tässä tapauksessa parempi, tosin toisaalta Silverlight-käyttöliittymässä on rajallinen määrä tilaa ruudulla kaikille työpäivityksille: Onko käytettävyyden kannalta parempi laittaa kaikki

tieto nähtäville pieneen tilaan, vai pakottaa käyttäjää hakemaan tieto erikseen näkyville? Tämä lieneekin makuasia. Käyttöliittymissä on eroja myös työpäivitysten suodattamisen suhteen. Toisin kuin Silverlight-käyttöliittymässä, ASP.NET-käyttöliittymässä valittaessa projektia tai työtä tehdään aina uusi HTTP-pyyntö palvelimelle. Lisäksi ASP.NET-käyttöliittymässä töistä voidaan näyttä joko ne työt, jotka eivät ole valmiita, tai ihan kaikki työt. Sen sijaan Silverlight-käyttöliittymässä töitä voidaan suodattaa kaikkien tilojen mukaan, joita on neljä erilaista. Tarkempi tilasuodatin olisi mahdollista toteuttaa myös ASP.NETillä, mutta kuitenkin myös tilasuodattimien käyttö aiheuttaa ylimääräisten HTTP-pyyntöjen lähettämisen, joka hidastuttaa käyttöä.

Viimeinen tarkasteltava toiminto on uusien asiakkaiden, projektien ja töiden lisääminen Työjonoon. ASP.NET-käyttöliittymässä sekä asiakkaiden että projektien lisääminen tehdään SharePointin omien luettelonäkymien kautta, koska kustomoitu ASP.NET-toteutus ei poikkeaisi SharePointin omista lomakkeista merkittävällä tavalla. Sen sijaan uusien töiden lisäämiselle on ASP.NET-käyttöliittymässä oma lomake. Tässä lomakkeessa käyttäjä valitsee projektin johon työ liitetään kaikkien Työjonoon liitettyjen projektien joukosta. Sen jälkeen käyttäjä täyttää lomakkeelle työn tiedot, liittää mahdollisen liitetiedoston ja tallettaa työn.

Silverlight-käyttöliittymässä nämä toiminnot on toteutettu kokonaan, jotta käyttäjän ei tarvitse siirtyä käyttöliittymästä pois lisätäkseen uusia tietoja Työjonoon. Itse tietojen lisääminen tapahtuu samaan tapaan kuin ASP.NET-käyttöliittymässä: Käyttäjä valitsee asiakkaan jolle uusi projekti luodaan, tai projektin jolle uusi työ luodaan, ja täyttää luotavan tietueen tiedot. Ainoa merkittävä ero toimintojen tekotavassa on siinä, että Silverlight-käyttöliittymässä käyttäjän on mahdollista suodattaa projekteja asiakkaiden mukaan, kun taas ASP.NET-käyttöliittymässä tätä mahdollisuutta ei ole. Toisaalta, tämä olisi mahdollista toteuttaa myös ASP.NETillä, mutta se vaatisi ylimääräisiä sivunlatauksia aina kun projekteja suodatetaan uusilla suodattimilla.

Kuten nähtiin, on Työjonon eri käyttöliittymien välillä erilaisia eroavaisuuksia käytettävyyden näkökulmasta. Osa näistä eroista on toiminnallisia, kun taas osa liittyy tietojen esitystapaan. Samalla huomattiin, että monet toiminnalliset eroavaisuudet olisivat

toteutettavissa molemmissa käyttöliittymissä, mutta tällöin ASP.NET-käyttöliittymässä jouduttaisiin tinkimään käytettävyydestä verrattuna Silverlight-käyttöliittymään. Joissakin tapauksissa tiettyjen toiminnallisuuksien toteuttaminen oli myös yksinkertaisesti helpompaa Silverlight-käyttöliittymässä johtuen teknologioiden välisistä eroavaisuuksista. Käyttöliittymien väliset käytettävyyserot on listattuna taulukossa 3.

Taulukko 3. Työjonon käyttöliittymien käytettävyyserot.

Toiminto	**Käytettävyysero**
Työpäivityksen lisääminen	Silverlight-käyttöliittymässä töitä voidaan suodattaa projektien mukaan. Työstä, johon päivitys lisätään, näytetään samalla tarkempia tietoja kuin ASP.NET-käyttöliittymässä.
Projektien ja töiden edistymisen seuranta	Silverlight-käyttöliittymässä suodattimien käyttö ei tarvitse sivun päivityksiä, ja on täten nopeampaa.
Projektien ja töiden edistymisen seuranta	Silverlight-käyttöliittymässä työpäivitykset pitää avata yksitellen toisin kuin ASP.NET-käyttöliittymässä.
Ajanvarausten tarkastelu	ASP.NET-käyttöliittymässä on käytössä kuukausikalenteri. Silverlight-käyttöliittymässä on viikko-kalenteri.
Ajanvarausten tarkastelu	Silverlight-käyttöliittymässä ajanvarauksesta saa lisätietoja ilman sivun päivitystä.
Ajanvarausten tarkastelu	Silverlight-käyttöliittymässä voi tarkastella useaa käyttäjää yhtäaikaa.
Ajanvarausten tarkastelu	Silverlight-käyttöliittymässä työpäivitykset pitää avata yksitellen. Työpäivityksille on olemassa tarkempita suodattimia.
Asiakkaiden, projektien ja töiden lisääminen	Silverlight-käyttöliittymässä projekteja voidaan suodattaa asiakkaan mukaan ja töitä voidaan suodataa projektin mukaan.
Kaikki	Silverlight-käyttöliittymän käyttö on verrattain nopeampaa, koska data voidaan ladata välimuistiin.

Mutta kuinka laadukas Työjonon käyttöliittymien käytettävyys on? Ottaen huomioon ISO:n käytettävyyden määritelmän, Työjonon käytettävyyden määrittäminen on sen

käyttäjistä kiinni [1]. Esimerkiksi toive Työjonon Silverlight-käyttöliittymän kaltaisesta kalenterinäkymästä tuli suoraan ohjelman loppukäyttäjiltä, joten tältä pohjalta voitaisiin todeta Silverlight-käyttöliittymän olevan ASP.NET-käyttöliittymää käytettävämpi. Koska osaa näistä toivotuista ominaisuuksista ei olisi voitu sellaisenaan tehdä ASP.NETillä voidaan myös todeta monipuolisten Internet-sovellusten teknologioiden, tässä tapauksessa Silverlightin, parantaneen Työjonon käytettävyyttä.

Tämä ei kuitenkaan tarkoita sitä, että Työjonon käytettävyys olisi välttämättä parhaimmillaan. Tällainen vaatisi käytettävyyden tarkkaa evaluoimista sekä loppukäyttäjien kanssa että käytettävyyden arviointimenetelmiä soveltaen. Puutteita Silverlight-käyttöliittymästäkin voidaan löytää helposti: Esimerkiksi osaan kalenterinäkymän toiminnoista, kuten työntekijöiden tarkempaan tarkasteluun, ei käyttöliittymä opasta millään tavalla. Sen sijaan käyttäjän tulisi itse etukäteen tietää, että kyseinen toiminto on olemassa. Koska Työjonoa käytetään vain sisäisesti rajatulla käyttäjäjoukolla ei tämä ole kriittinen puute, mutta se on kuitenkin olemassa oleva käytettävyysongelma jota ei voida kiistää.

Kaikki Työjonon kehityksessä tehdyt käytettävyyteen liittyvät muutokset tehtiin käyttäjäpalautteen pohjalta. Toisin sanoen, Työjonon kehityksessä ei olla otettu huomioon kirjallisuudessa esitettyjä keinoja käytettävyyden kehittämiseen. Tämän takia Työjonon osalta ei ole mahdollista pohtia miten hyvin teoreettiset keinot käytettävyyden kehittämiseen oikeasti parantavat ohjelman käytettävyyttä. Lisäksi, spekulatiivinen teoreettisen keinojen vaikutusten pohdiskelu ei ole mielekästä, sillä kirjallisuudessa esitetyt keinot tai lähtökohdat eivät kerro suoraan miten käytettävyyttä tulisi kehittää. Sen sijaan ne on tarkoitettu sovellettaviksi tapauskohtaisesti. Esimerkiksi älykkäät käyttöliittymät, joista Jaquero et al. kirjoittivat, on esitetty mahdollisena tapana kehittää käytettävyyttä, mutta niiden osalta ei suoraan sanota mitä älykkäiden käyttöliittymien tulisi tehdä [19]. Vastaavasti Shneidermanin esittämät kahdeksan kultaista sääntöä ovat varsin ympäripyöreitä, eikä esimerkiksi ohjetta ”Tavoittele yhtenäisyyttä” [16] voi hyödyntää käytännössä soveltamatta sitä kehitettävän ohjelmiston ja sen käyttäjien tarpeiden mukaisesti.

Mikäli Työjonon nykyistä käytettävyyttä tarkasteltaisiin kirjallisuudessa esitettyjen käytettävyysohjeiden valossa, olisivat kvantitatiiviset ohjeet ainoita joita voitaisiin hyödyntää suoraan. Tämä siksi, että niitä on mahdollista tarkastella ja hyödyntää ilman tapauskohtaista soveltamista. Esimerkiksi ohje ”Käyttöliittymässä tulisi olla neljää eri väriä” on yksiselitteinen ja käytettävissä sellaisenaan kun tarkastellaan Työjonon tai jonkin muun sovelluksen käytettävyyttä. Toisaalta, tämänlainen ohje ei ota kantaa siihen minkälaisia käytettävien värien tulisi olla, joten erilaisista käyttöliittymistä, joissa on käytössä yhtä monta väriä, voidaan saada hyvinkin erilaisia tuloksia riippuen valituista väreistä. Lisäksi, kuten Ivory ja Megraw tutkimuksessaan totesivat, eivät kvantitatiiviset ohjeet välttämättä korreloi subjektiivisesti koetun, hyvän käytettävyyden kanssa [20]. Tästä johtuen myös kvantitatiivisiin ohjeisiin perustuva, Työjonon käytettävyyden laatuun liittyvä spekulatiivinen pohdiskelu ei ole erityisen mielekästä, eikä se tuottaisi tuloksia jotka varmasti pätisivät myös käytännössä.

7 KESKUSTELU

Kuten luvussa 2 todettiin, käytettävyyden arvioimiselle ja kehittämiselle on olemassa useita erilaisia tapoja. Nämä tavat lähestyvät käytettävyyttä joko objektiivisesti tai subjektiivisesti [14]. Objektiiviset tavat pyrkivät kertomaan miten käytettävyyttä tulisi käsitellä, kun taas subjektiiviset tavat ottavat huomioon ohjelmien loppukäyttäjien tai muiden subjektiivisten arvioijien mielipiteet. Tutkimustyön kannalta voidaan pitää harmillisena sitä, että objektiiviset käytettävyysohjeet eivät todistetusti aina korreloidu subjektiivisten käytettävyyskokemusten kanssa [20]. Harmillista tämä on siksi, että objektiivisia kohteita on helpompi tutkia kuin subjektiivisia kohteita, ja subjektiivisista kokemuksista voi olla vaikeampi tehdä päteviä yleistyksiä.

Johtuen edellä mainitusta objektiivisten ja subjektiivisten lähtökohtien ristiriidasta [20], kirjallisuudessa esitettyjä objektiivisia käytettävyyden kehittämisen keinoja on vaikea esittää yleispätevinä ratkaisuina käytettävyyden parantamiseksi. Toisaalta, näitä keinoja ei myöskään kannata sulkea pois vain siksi, että ne ovat objektiivisia. Jonkin käytettävyyden kehittämisen keinon objektiivisuus ei sulje pois sitä mahdollisuutta, että jokin käyttäjä kokisi sen subjektiivisesti hyväksi keinoksi. Tästä lähtökohdasta katsottuna objektiivisten käytettävyyteen liittyvien keinojen ja ohjeiden tutkiminen voi olla hyödyllistä, kunhan muistetaan, että niiden käyttö tulisi pohjautua subjektiivisista kokemuksista saatuun palautteeseen. Näin käytettynä objektiivisista käytettävyyden kehittämisen ja arvioimisen keinoista saataisiin koostettua joukko potentiaalisesti hyödyllisiä työkaluja. Tällaisia työkaluja ovat esimerkiksi älykkäät käyttöliittymät [19], mielenkiintoiset hetket [17], Shneidermanin kahdeksan kultaista sääntöä [16] sekä TOTE-mallin soveltaminen [7].

Monipuolisten Internet-sovellusten (RIA) teknologioilla todettiin olevan mahdollista kehittää SharePoint-sovellusten käytettävyyttä niin teorian kuin käytännönkin näkökulmasta. Teoreettisesta näkökulmasta tarkasteltuna RIA-teknologiat paljastuivat mahdollistaviksi teknologioiksi, joiden avulla voidaan toteuttaa käyttöliittymiä jollaisia HTML:llä ei voida tehdä. Useita objektiivisia käytettävyyden kehittämisen keinoja, kuten mielenkiintoisia hetkiä, olisikin erittäin vaikeaa tai jopa mahdotonta toteuttaa ilman RIA-teknologioita [17].

Työjono-ohjelmiston valossa monipuolisten Internet-sovellusten teknologioilla todettiin olevan sekä mahdollistava että teknologian ominaisuuksiin pohjautuvia vaikutuksia käytettävyyden kannalta. RIA-teknologiat mahdollistivat Työjonon Silverlight-käyttöliittymässä toteutetun viikkokalenterin toteuttamisen nykyisessä muodossaan. Toisaalta, Työjonon Silverlight-käyttöliittymän käytön nopeus, verrattuna Työjonon ASP.NET-käyttöliittymään, perustui pelkästään Silverlightin teknisiin ominaisuuksiin, erityisesti esitettävän datan lataamiseen sovelluksen välimuistiin. Tässä valossa voidaan todeta, että RIA-teknologioita on mahdollista käyttää parantamaan SharePoint-sovellusten käytettävyyttä sekä mahdollistavana teknologiana, että niiden ominaisten teknisten ominaisuuksien puolesta.

Työjono-ohjelmiston tapauksesta nähtiin, että sellaisen käytettävyyden toteuttaminen, jonka ohjelmiston loppukäyttäjät kokevat laadukkaaksi, vaatii kyseisten loppukäyttäjien osallistumista ohjelmiston suunnitteluun ja testaukseen. Työjonon lopullisessa käyttöliittymässä toteutettu viikkokalenteri tehtiin pelkästään loppukäyttäjiltä saadun palautteen johdosta. Mikäli Työjonon toteutuksessa olisi hyödynnetty objektiivisia käytettävyyden arviointimetodeita subjektiivisen testauksen sijaan, olisi tämä kyseinen ominaisuus jäänyt toteuttamatta. Toisaalta, subjektiiviset arvioijat näkevät vain niitä käytettävyysongelmia, jotka he itse kokevat merkittäviksi. Tästä johtuen, esimerkiksi, Työjonon Silverlight-käyttöliittymä ei välttämättä ole intuitiivinen käyttää, sillä sen loppukäyttäjät eivät kokeneet intuitiivisuuden parantamista oleelliseksi.

Työjonon pohjalta voidaankin todeta, että SharePoint-sovellusten käytettävyyden kehittämisessä tulisi ottaa huomioon sekä subjektiiviset mielipiteet että objektiiviset ohjeet. Subjektiiviset mielipiteet kertovat käyttäjien todellisista toiveista ohjelman suhteen, kun taas objektiiviset ohjeet voivat paljastaa käytettävyysongelmia joita subjektiiviset arvioijat eivät olleet huomanneet. Näiden molempien keinojen käyttäminen on kannattavaa erityisesti silloin, kun subjektiivisten testaajien joukko ei vastaa kaikkia SharePoint-sovelluksen lopullisia käyttäjiä. Teoreettisen tutkimuksen pohjalta tehty johtopäätös objektiivisista keinoista mahdollisesti hyödyllisinä työkaluina käy yhteen tämän havainnon kanssa. Kirjallisuudesta löydettyjä käytettävyyden kehittämisen keinoja kannattaneekin

hyödyntää sen jälkeen, kun kehitettävässä SharePoint-sovelluksessa on hyödynnetty subjektiivisia arvioijia. Tällöin subjektiivisten arvioijien avulla löydettäisiin sovelluksessa olevat, käyttäjille merkittävät käytettävyysongelmat, ja objektiivisilla keinoilla käytettävyyttä viimeisteltäisiin.

Avoimeksi kysymykseksi jää, ovatko jotkut kirjallisuudessa esitetyt objektiiviset käytettävyyden kehittämisen keinot sellaisia, että ne subjektiivisesti tarkasteltuna aina tai lähes aina parantavat käytettävyyttä? Mikäli tällaisia keinoja on olemassa, olisi niiden löytäminen hyödyllistä. Tällöin niitä voitaisiin hyödyntää yleisesti ilman erillistä subjektiiviseen tarkasteluun perustuvaa verifiointia. Näihin objektiivisiin keinoihin keskittyvä jatkotutkimus, jossa näitä keinoja arvioitaisiin nimenomaan subjektiivisesti, voisikin olla tarpeen.

Tässä työssä tehdyn kyselytutkimuksen valossa voidaan todeta, että käytettävyys koetaan merkittäväksi ominaisuudeksi SharePoint-sovellusten kohdalla, ja että sen kehittämiselle SharePoint-sovelluksissa on tarvetta. Kuitenkin johtuen kyselyn pienestä otannasta tätä tulosta ei voida yleistää, eikä muitakaan kyselyssä saatuja tuloksia voida pitää yleisesti pätevinä. Tästä johtuen käytettävyyden kokemiseen liittyvälle lisätutkimukselle on tarvetta, jotta käytettävyyden merkitystä sovellusten käyttäjille saadaan kartoitettua tarkemmin. Lisäksi tutkimukset, joissa käytettävyys asetetaan vastakkain muiden ohjelmistoihin liitettyjen ominaisuuksien kanssa, voisivat tuottaa dataa, jonka valossa käytettävyyden kehitystä voitaisiin ohjelmistoprojekteissa priorisoida paremmin.

8 YHTEENVETO

Tässä työssä tutkittiin SharePoint-sovellusten käytettävyyttä ja sen kehittämistä. Käytettävyyttä sekä sen arvioimista ja kehittämistä käsiteltiin kirjallisuustutkimuksen pohjalta. Käytettävyydelle löydettiin erilaisia teoreettisia kehitystapoja. Kuitenkin nämä kirjallisuudessa esitetyt tavat olivat enimmäkseen objektiivisia, joten ne eivät ottaneet huomioon sovellusten käyttäjien subjektiivista kokemusta käytettävyydestä.

Monipuolisten Internet-sovellusten vaikutusta SharePoint-sovellusten käytettävyyteen selvitettiin sekä teorian että käytännön ohjelmistotototeutuksen pohjalta. Sekä teorian että käytännön tulosten pohjalta monipuolisten Internet-sovellusten teknologiat ovat enimmäkseen mahdollistavia teknologioita, joiden avulla voidaan toteuttaa kompleksisempia käyttöliittymiä. Ne tarjoavat parempaa käytettävyyttä HTML-pohjaisiin käyttöliittymiin verrattuna myös tiettyjen teknisten ominaisuuksiensa ansiosta.

Tämä työ pohjautui SharePoint-ohjelmistotototeutukseen, jossa toteutettavalle ohjelmistolle luotiin kaksi erilaista käyttöliittymää käytettävyyden takia. Toteutuksen jälkeen todettiin, että ohjelmistossa päädyttiin sen käyttäjien mielestä hyvään käytettävyyteen, koska ohjelmistoa kehitettiin kyseisiltä käyttäjiltä saadun palautteen pohjalta. Kuitenkin objektiivisten mittareiden mukaan ohjelmiston käytettävyys jäi puutteelliseksi, ja käytettävyyden todettiin olevan epätäydellistä. Tästä pääteltiin, että subjektiivinen testaaminen on tärkeää, jotta ohjelmistoon saadaan lisättyä käyttäjille välttämättömiä käytettävyysominaisuuksia. Lisäksi objektiivisten ohjeiden noudattaminen on tärkeää, jotta käytettävyydessä voidaan ottaa huomioon asioita, jotka testaajilta jäävät huomaamatta.

Tässä työssä tehtiin kyselytutkimus SharePointin käytettävyydestä ja käytettävyyden merkityksestä SharePoint-sovelluksissa. Kysely lähetettiin yrityksiin, joissa SharePointia käytetään työnteon välineenä. Vastauksia kyselyyn saatiin vain seitsemän kappaletta, joten sen otos jäi erittäin pieneksi. Pienestä otoksesta johtuen kyselyn tuloksista ei voitu tehdä merkittäviä yleistyksiä, mutta saaduista vastauksista kuitenkin nähtiin, että käytettävyys koetaan merkittäväksi ominaisuudeksi ja että sen kehittämiselle SharePoint-sovelluksissa on tarvetta.

LÄHTEET

1. ISO 9241-11: Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs): Part 11: Guidance on Usability, International Organization For Standardization, 1998.
2. Macleod, M., Usability: practical methods for testing and improvement, *Proceedings of the Norwegian Computer Society Software '94 Conference*, Sandvika, Norway, 1994.
3. Survey Reveals Microsoft SharePoint 2010 Deployments Increased Five-fold in 6 Months, Vocus PRW Holdings, LLC, viitattu 26.5.2012, http://www.prweb.com/releases/2011/2/prweb8121899.htm.
4. Petrie, H., Kheir, O., The relationship between accessibility and usability of websites, *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2007)*, ACM Press, April 28 - May 3, San Jose, CA, USA, 2007, pp.397-406.
5. Pühretmair, F., Miesenberger, K., Making Sense of Accessibility in IT Design - Usable Accessibility vs. Accessible Usability, *DEXA Workshops 2005*, 2005, pp. 861–865.
6. Cockton, G., From quality in use to value in the world, *CHI '04 extended abstracts on Human factors in computing systems (CHI EA '04)*, ACM Press, April 24-29, Vienna, Austria, 2004, pp. 1287-1290.
7. Miller, G.A., Galanter, E., Pribram, K.H., Plans and the structure of behavior, Holt, Rhinehart, & Winston., USA, 1960.
8. Shneiderman, B., Universal Usability, *Communications of the ACM*, Vol. 43, ACM Press, USA, 2000, pp. 84-91.
9. van Welive, M., van der Veer, G.C., Eliëns, A., Breaking down Usability, *Proceedings of INTERACT '99*, August 30 – September 3, Edinburgh, Scotland, 1999, pp. 613-620.
10. Shneiderman, B., Promoting universal usability with multi-layer interface design, *Proceedings of the 2003 conference on Universal usability (CUU '03)*, ACM Press, November 10-11, Vancouver, Canada, 2003, pp. 1-8.

11. Nielsen, J., Mack, R.L., Usability Inspection Methods, John Wiley & Sons, USA, 1994.
12. Kjeldskov, J., Stage, J., New techniques for usability evaluation of mobile systems, *International Journal of Human-Somputer Studies* Vol. 60, Elsevier, 2004, pp. 599-620.
13. Hollingsed T., Novick, D.G., Usability inspection methods after 15 years of research and practice, *Proceedings of the 25th annual ACM international conference on Design of communication (SIGDOC '07)*, ACM Press, October 22-24, El Paso, Texas, USA, 2007, pp. 249-255.
14. Greenberg, S., Buxton, B., Usability Evaluation Considered Harmful (Some of the Time), *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI '08)*, ACM Press, April 5-10, Florence, Italy, 2008, pp. 111-120.
15. Sajedi, A., Mahdavi, M., Pourshirmohammadi, A., Nejad, M.M., Fundamental Usability Guidelines for User Interface Design, *International Conference on Computational Sciences and Its Applications (ICCSA '08)*, June 30 - July 3, 2008, pp. 106-113.
16. Shneiderman, B., Plaisant, C., Designing the user interface: Strategies for effective Human-Computer Interaction, 4th edition, Addison Wesley, USA, 2004.
17. Scott, B., Theresa, N., Designing Web Interfaces, 1st edition, O'Reilly Media, Inc., USA, 2009.
18. Gulz, A., Spatially Oriented and Person Oriented Thinking—Implications for User Interface Design, *Education and Information Technologies*, Vol. 7, Springer Netherlands, 2002, pp. 67-80.
19. Jaquero, V.L., Montero, F., Molina, J.P., Gonz´lez, P., Intelligent User Interfaces: Past, Present and Future, *Engineering the User Interface*, Springer London, 2009, pp. 259-270.
20. Ivory, M.Y., Megraw, R., Evolution of web site design patterns, *ACM Transactions on Information Systems (TOIS)*, Vol. 23, ACM Press, New York, NY, USA, 2005, pp. 463-497.
21. Shneiderman B., Designing for fun: how can we design user interfaces to be more fun?, *interactions*, Vol. 11, ACM Press, New York, NY, USA, 2004, pp. 48-50.

22. Wahlster, W., Maybury, M., An Introduction to Intelligent User Interfaces, *Readings in Intelligent User Interfaces (RUIU)*, Morgan Kaufmann Publishers, Inc., San Francisco, USA, 1998, pp. 1- 13.
23. Boiko, B., Content Management Bible, 2nd edition, Wiley Publishing, Inc., USA, 2005.
24. Fisher, P.T., Duskis, S., Using Spring with a Content Management System, Spring Persistence, Apress, 2009, pp.163-185.
25. Chieu, T.C., Liangzhao Z., Mohindra, A., An extensible enterprise content management system with Service Component Architecture, *IEEE International Conference on Service Operations and Logistics, and Informatics, 2008 (IEEE/SOLI 2008)*, Vol. 1, IEEE, Beijing, China, 2008, pp. 1131-1137.
26. Tyrväinen, P., Päivärinta, T., Salminen, A., Iivari, J., Characterizing the evolving research on enterprise content management, *European Journal of Information Systems*, Vol. 15, 2006, pp. 627–634.
27. EMC Documentum Standard, EMC Corporation, viitattu 17. 5. 2012, http://www.emc.com/products/family/documentum-standard-family.htm.
28. Laserfiche Avante, Laserfische, viitattu 17. 5. 2012, http://www.laserfiche.com/en-us/Products/Avante.
29. OpenText ECM Suite 2010, OpenText Corporation, viitattu 17. 5. 2012, http://www.opentext.com/2/global/products/products-opentext-ecm-suite.htm.
30. Lappin, J., McLeod, J., Investigation into the use of Microsoft SharePoint in Higher Education Institutions, School of Computing, Engineering & Information Sciences, Northumbria University, 2010.
31. Drupal project Frequently Asked Questions, Drupal.org, viitattu 17. 5. 2012, http://drupal.org/drupal-faq.
32. What is Joomla?, Open Source Matters, Inc., viitattu 17. 5. 2012, http://www.joomla.org/about-joomla.html.
33. What is Titan CMS?, Northwoods Software Development, Inc., viitattu 17. 5. 2012, http://titancms.com/AboutUs.htm.
34. McKenna, E., Laahs, K., Vanamo, V.M., Microsoft SharePoint 2010 All-In-One for Dummies, Wiley Publishing, Inc., USA, 2011.

35. Lai, L., Turban, E., Groups Formation and Operations in the Web 2.0 Environment and Social Networks, *Group Decision and Negotiation*, Vol. 17, Springer Netherlands, 2008, pp. 387-402.
36. Conconi, A., Gusmeroli, S., Ratti, R., A Collaboration Platform For Pvc, *Methods and Tools for Collaborative Networked Organizations*, Springer US, 2008, pp. 307-334.
37. Terracina, A., Beco, S., Grenham, A., Le Duc, I., Rossi, A., Fusco, L., Architecture of the DICTE Collaboration Platform, *Lecture Notes in Computer Science*, Vol. 5738, 2009, Springer Berlin / Heidelberg, pp. 261-268.
38. Rooksby, J., Sommerville, I., The Management and Use of Social Network Sites in a Government Department, *Computer Supported Cooperative Work (CSCW)*, Springer Netherlands, 2011, pp. 1-19.
39. Brzozowski, M.J, WaterCooler: Exploring an organization through enterprise social media, *Proceedings of the ACM 2009 international conference on Supporting group work (GROUP '09)*, ACM Press, May 10-13, Sanibel Island, FL, USA, 2009, pp. 219-228.
40. DiMicco, J., Millen, D.R., Geyer, W., Dugan, C., Brownholtz, B., Muller, M., Motivations for Social Networking at Work, *Proceedings of the 2008 ACM conference on Computer supported cooperative work (CSCW '08)*, ACM Press, November 8-12, San Diego, CA, USA, 2008, pp. 711-720.
41. Romeo, P., The D Street Case Study, *Workshop on Social Networking in Organizations (CSCW '08)*, November 8-12, San Diego, CA, USA, 2008.
42. Lotus Notes Overview, International Business Machines Corporation, viitattu 17. 5. 2012, http://www-01.ibm.com/software/lotus/products/notes/.
43. Oracle WebCenter, Oracle Corporation, viitattu 17. 5. 2012, http://www.oracle.com/us/products/middleware/webcenter/overview/index.html.
44. Liferay Portal Product Overview, Liferay Inc., viitattu 17. 5. 2012, http://www.liferay.com/products/liferay-portal/overview.
45. Google Apps, Google Inc., viitattu 9. 6. 2012, http://www.google.com/apps/intl/fi/group/index.html.
46. Microsoft Office 365, Microsoft Corporation, viitattu 9. 6. 2012, http://www.microsoft.com/businessproductivity/fi/fi/products/office-365.aspx

47. Bozzon, A., Comai, S., Fraternali, P., Carughi, G.T., Conceptual Modeling and Code Generation for Rich Internet Applications, *Proceedings of the 6th international conference on Web engineering (ICWE'06)*, ACM, July 11-14, Palo Alto, CA, USA, 2006, pp. 353-360.
48. Komischke, T., The Impact of Rich Application Frameworks on User Experience Design, *Lecture Notes in Computer Science*, Vol. 6761, Springer Berlin / Heidelberg, 2011, pp. 92-97.
49. Flanagan, D., JavaScript: The Definitive Guide 6th edition, O'Reilly Media, Inc., USA, 2011.
50. Niemeyer, P., Knudsen, J., Learning Java, 3rd edition, O'Reilly Media, Inc., USA, 2005.
51. Finkelstein, E., Leete, G., Flash CS4 for Dummies, Wiley Publishing, Inc., USA, 2008.
52. Ghoda, A., Introducing Silverlight 4, Apress, USA, 2010.
53. Garrett, J.J., Ajax: A new approach to Web applications, Adaptive Path, USA, 2005.
54. David M., HTML5: Designing Rich Internet Applications, Elsevier, Inc., USA, 2010.
55. Myers B.A., Rosson, M.B., Survey on user interface programming, *Proceedings of the ACM CHI conference on Human factors in computing systems (CHI '92)*, ACM, June 3-7, Monterey, CA, USA, 1992, pp. 195-202.
56. Krause, J., Langhirt, C., Sterff, A., Pehlke, B., Döring, M., SharePoint 2010 as a Development Platform, Apress, USA, 2010.
57. Preciado, J.C., Linaje, M., Comai, S., Sanchez-Figueroa, F., Designing Rich Internet Applications with Web Engineering Methodologies, *9th IEEE International Workshop on Web Site Evolution (WSE 2007)*, IEEE, October 5-6, 2007, pp.23-30.
58. Korhonen, H., Holm, J., Heikkinen, M., Utilizing Sound Effects in Mobile User Interface Design, *Lecture Notes in Computer Science*, Vol. 4662, Springer Berlin / Heidelberg, 2007, pp. 283-296.

59. Book, M., Brückmann, T., Gruhn, V., Hülder, M., A Conceptual Framework for User Input Evaluation in Rich Internet Applications, *Lecture Notes in Computer Science*, Vol. 5648, Springer Berlin / Heidelberg, 2009, pp 275-282.
60. Tesarik, J., Dolezal, L., Kollmann, C., User interface design practices in simple single page web applications, *First International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2008)*, IEEE, August 4-6, 2008, pp. 223-228.
61. Bates, S., Smith, T., SharePoint 2010 User's Guide, Apress, USA, 2010.
62. Using a records archive versus managing records in place (SharePoint Server 2010), TechNet, Microsoft Corporation, viitattu 26. 5. 2012, http://technet.microsoft.com/en-us/library/ee424394.aspx.
63. Protalinski, E., Microsoft Lync 2010 arrives, succeeds Office Communicator, Ars Technica, viitattu 20. 5. 2012, http://arstechnica.com/information-technology/2010/11/microsoft-lync-2010-released-succeeds-office-communicator/.
64. Esposito, Dino, Programming Microsoft ASP.NET 2.0 Applications: Advanced Topics, Microsoft Press, USA, 2006.
65. Krishnan, M., Beadle, P., Silverlight 4 for Dummies, Wiley Publishing, Inc., USA, 2010.
66. Exchange Server, Microsoft Corporation, viitattu 25. 5. 2012, http://www.microsoft.com/exchange/en-us/exchange-server.aspx.

LIITE 1. Asiakaskyselyyn saadut vastaukset

Vastaaja 1.

Kysymys	Vastaus
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-loppukäyttäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Tiedostojen jakoon, aktiiviseen työstämiseen, informaatiokanavana.
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Intranet
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Nykyaikainen työkalu, jolla kollaboraatiota voidaan oikeasti harrastaa.
”Miten SharePointin käyttö on vastannut odotuksianne?”	Teknisiltä ominaisuuksiltaan hyvin, käyttäjien aktiivisuus vain pitäisi saada nostettua ylemmäksi.
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	5
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	5
”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	4
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	4
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Vanhoista työskentelytavoista luopuminen ja uusien omaksuminen.
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Kyllä

Vastaaja 2.

Kysymys	Vastaus
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-loppukäyttäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Dokumenttien hallintaan ja kollaborointiin
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Intranet
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Odotin että se on erittäin rasks ohjelmisto
”Miten SharePointin käyttö on vastannut odotuksianne?”	Se on erittäin raskas ohjelmisto
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	5
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	5

(jatkuu)

”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	3
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	3
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Se hidastaa konetta
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Kyllä

Vastaaja 3.

Kysymys	**Vastaus**
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-loppukäyttäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Intrasivustot toteutettu konsernissa. Omassa toiminnossamme: sovellusympäristönä käytössä (johdon työpöytä), teossa työlistoja, muutoshallintatyökalu jne
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Intranet
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Että SP olisi mahdollisimman laaja-alainen "mahdollistaja" erilaisille tarpeille. Toisaalta valmiita toiminnallisuuksia ja toisaalta mahdollisuus räätälöidä.
”Miten SharePointin käyttö on vastannut odotuksianne?”	Hyvin - SP on mahdollistanut halutut toiminnallisuudet varsin pitkälle. Varsinaisia "pettymyksiä" on ollut vähän. SP on kuitenkin niin laaja kokonaisuus että omaksuminen organisaatiossamme on vasta aluillaan - tämä on toki osaltaan myös haaste jonka SP tuo tullessaan (mammutti johon "porautuminen" ottaa aikansa)
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	5
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	5
”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	4
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	4
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Suorituskykyhaasteet intranet-toteutuksessa (en itse tunne niitä tarkasti).
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Kyllä

(jatkuu)

Vastaaja 4.

Kysymys	Vastaus
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-ylläpitäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Tiedostojen jakeluun.
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Intranet pienemmälle yksikölle
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Ei vastausta
”Miten SharePointin käyttö on vastannut odotuksianne?”	Ei vastausta
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	5
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	5
”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	3
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	4
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Koulutuksen puute : ei osata rakentaa/hyödyntää ominaisuuksia
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Kyllä

Vastaaja 5.

Kysymys	Vastaus
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-loppukäyttäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Käytämme sharepointia pääasiassa kollaborointiin ja tiedostojen jakeluun, mutta lisääntyvässä määrin sovellusympäristönä esimerkiksi muutoksenhallinnassa.
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Ei vastausta
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Odotuksina on: Joustavasti konfiguroitava, helppokäyttöinen, edullinen.
”Miten SharePointin käyttö on vastannut odotuksianne?”	Nykyinen 2007 versio ei ole vastannut täysin odotuksiin, mutta joihinkin asioihin parannusta on luvassa 2010 versiossa.
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	5
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	5

(jatkuu)

”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	3
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	4
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Liian helppo tehdä uutta nopeasti ajattelematta -> sivujen ja data määrä kasvaa nopeasti, jolloin käyttäjä usein tuskastuu oikean sivuston etsimiseen. Osittain myös organisaation tiettyjen osien kankeus on vaikeuttanut toimintaa.
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Kyllä

Vastaaja 6.

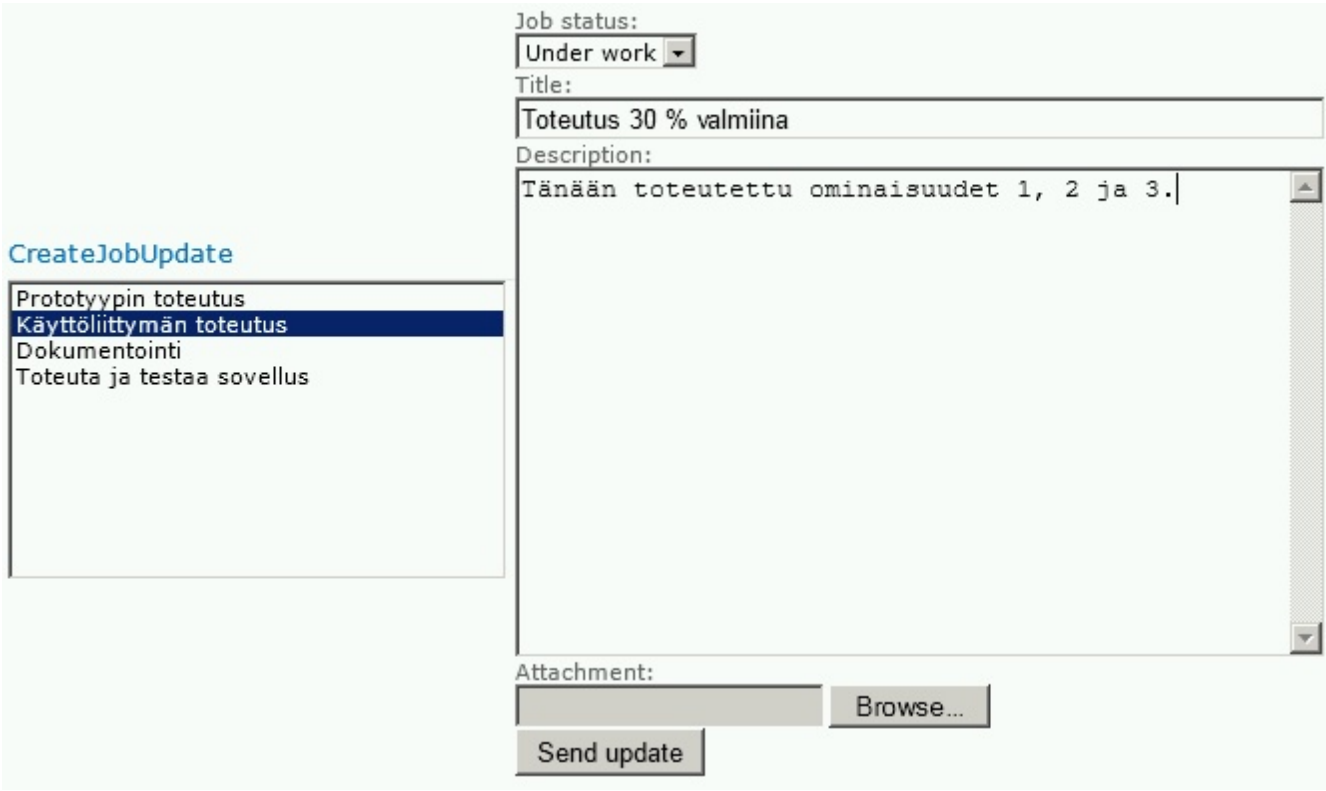
Kysymys	**Vastaus**
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-ylläpitäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Web-sivuihin, sovellusympäristönä, tiedostojenjakeluun.
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Yhdelle yritykselle internet-sivusto, extranet kahdelle yritykselle ja sisäisiä palveluja tietyille kohderyhmille organisaation sisällä (ei vielä intranet).
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	SharePoint tuotteena petrannut edellisestä versiosta.
”Miten SharePointin käyttö on vastannut odotuksianne?”	Mikäli out-of-the-box -toiminnoilla pärjää tällöin saattaa olla kustannnustehokasta. Räätälöimällä kuluu rahaa ja aikaa.
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	4
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	4
”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	3
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	3
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Tuotteen laajat mahdollisuudet ja kokonaisuuden johtaminen.
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	En osaa sanoa

(jatkuu)

Vastaaja 7.

Kysymys	Vastaus
”Mikä seuraavista vastaa sinua parhaiten?”	Olen SharePoint-ylläpitäjä
”Mihin tarkoituksiin käytätte tai suunnittelette käyttävänne SharePointia?”	Websivusto, työtila-ratkaisut
”Mitkä seuraavista verkoista on teidän organisaatiossanne toteutettu SharePointilla?”	Intranet, extranet, internet (tekeillä)
”Minkälaisia odotuksia teillä on, tai oli, SharePointin käytön suhteen?”	Odotusarvo oli, että OOB:lla pääsee hyvinkin pitkälle verkkoratkaisuissa - toimittajien SP-osaaminen ei välttämättä aina osoittautunut odotusten mukaiseksi.
”Miten SharePointin käyttö on vastannut odotuksianne?”	Alustaan liittyvät ongelmat ovat haitanneet itse sovelluksista nauttimista...
”Kuinka tärkeänä pidät tietokonesovellusten käytettävyyttä?”	5
”Kuinka tärkeänä pidät SharePoint-sovellusten käytettävyyttä?”	5
”Kuinka tyytyväinen olet SharePointin käytettävyyteen?”	3
”Kuinka tärkeänä pidät SharePoint-sovellusten visuaalista ilmettä?”	4
”Mitkä ovat teillä olleet suurimmat haasteet SharePointin käytössä?”	Versionvaihdos 2007:sta ja usean eri toimijan tuottamat sovellukset samalla alustalla. Toimijoiden kommunikaatio.
”Onko teillä työasemissa käytettävissä Silverlight-laajennus?”	Ei

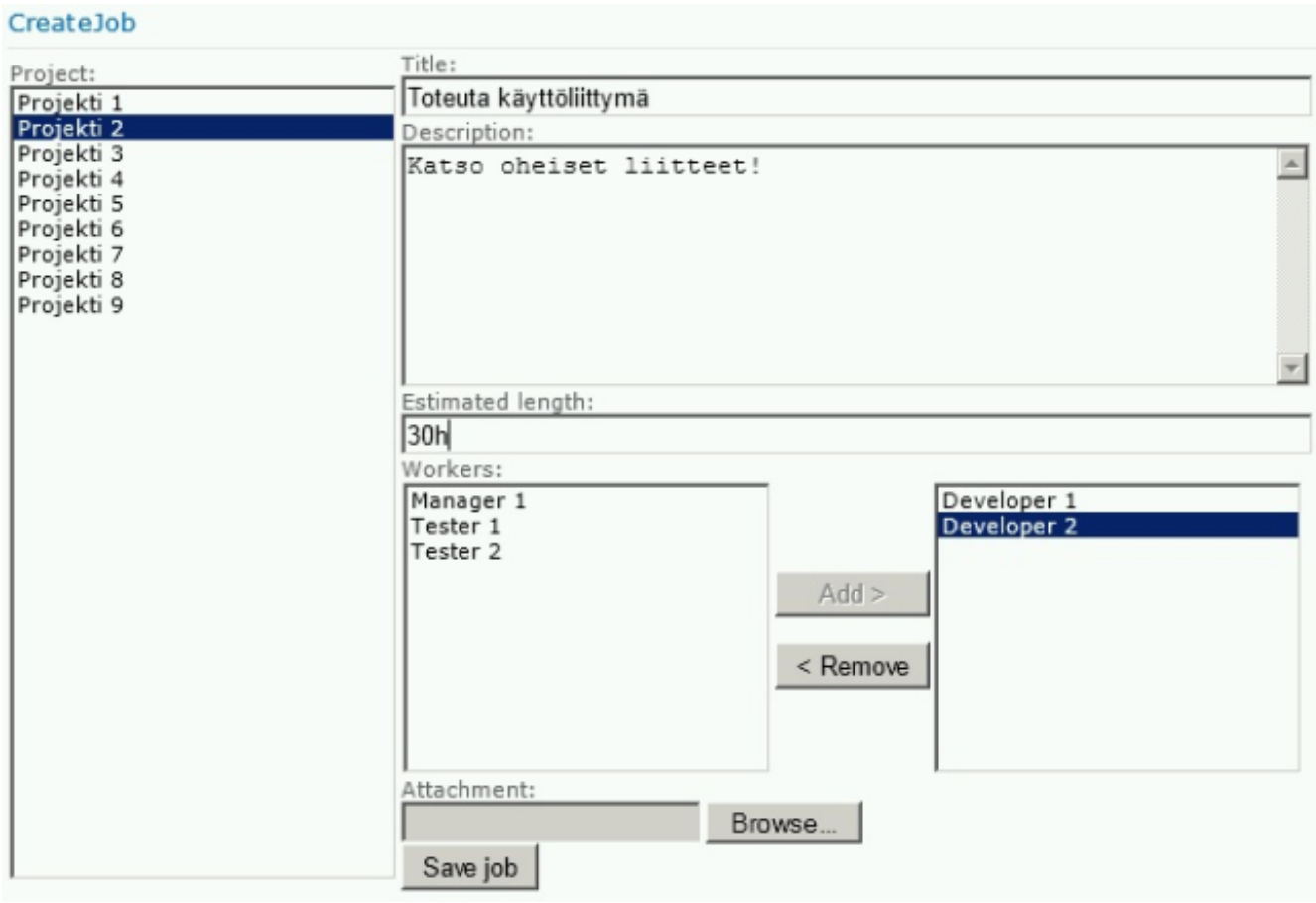
LIITE 2. Työjonon ASP.NET-käyttöliittymän kuvankaappaukset



Kuva: Projektipäivityksen lisäämisen ASP.NET-web-osa

(jatkuu)

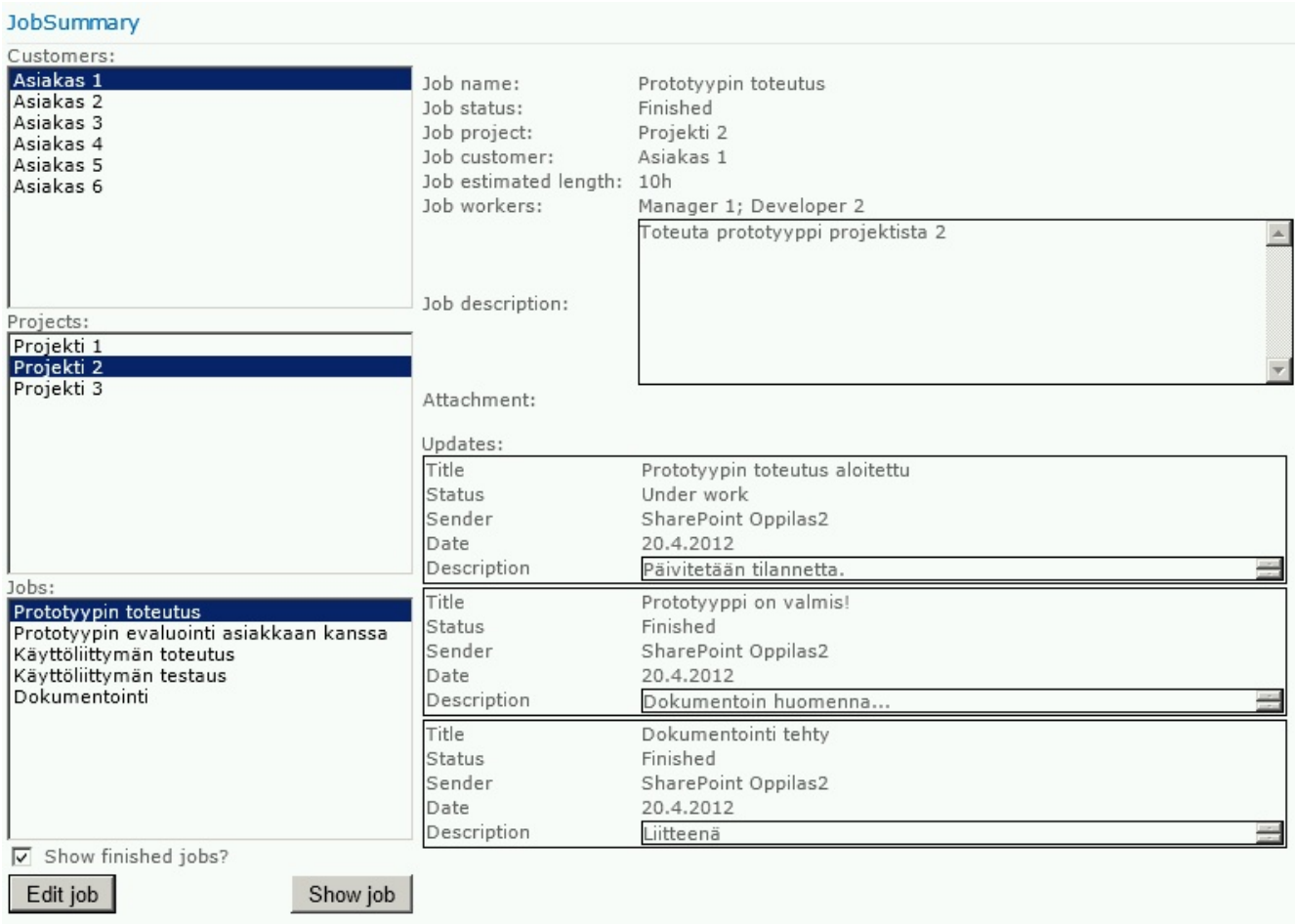
LIITE 2. (jatkoa)



Kuva: Työtehtävien lisäämisen ASP.NET-web-osa

(jatkuu)

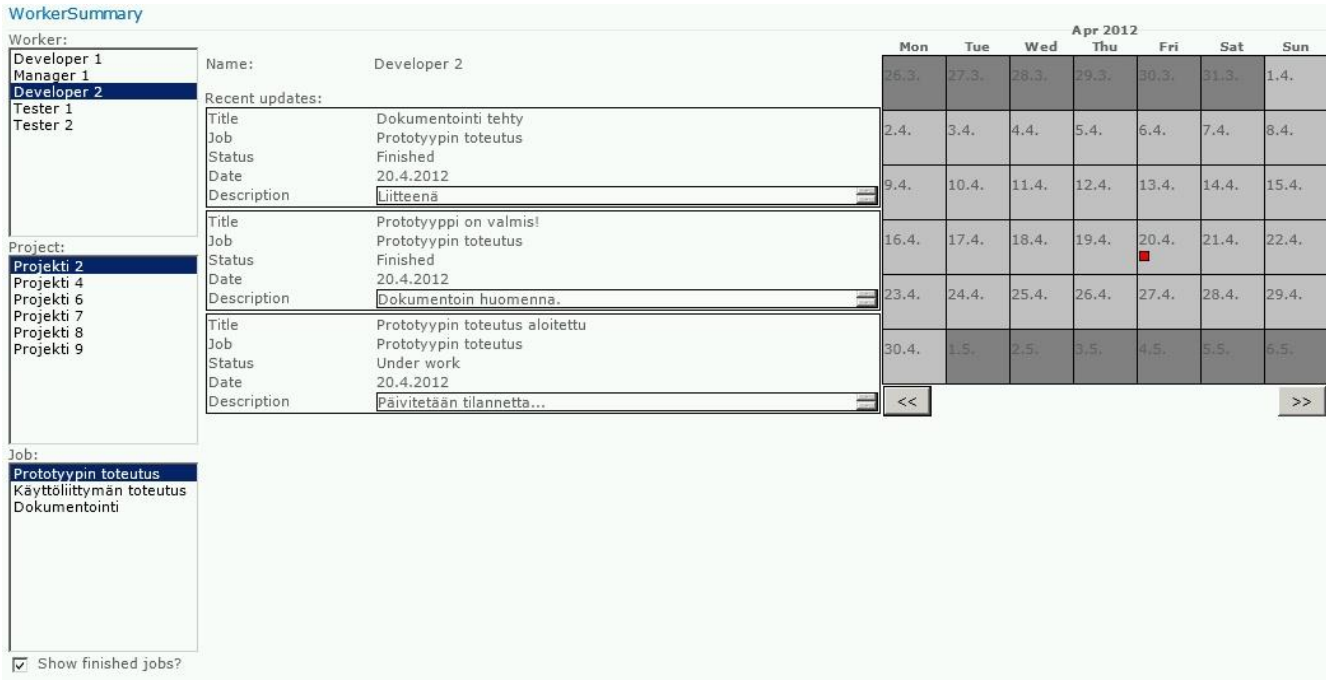
LIITE 2. (jatkoa)



Kuva: Projektinseurannan ASP.NET-web-osa

(jatkuu)

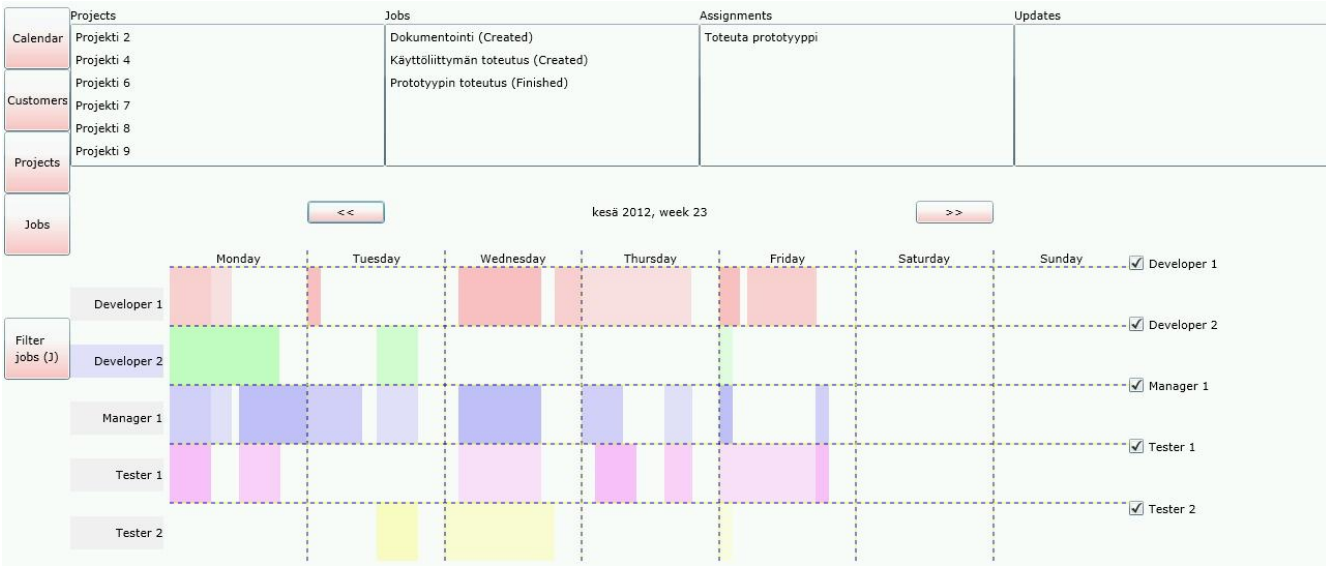
LIITE 2. (jatkoa)



Kuva: Työntekijän seurannan ASP.NET-web-osa

LIITE 3. Työjonon Silverlight-käyttöliittymän kuvankaappaukset

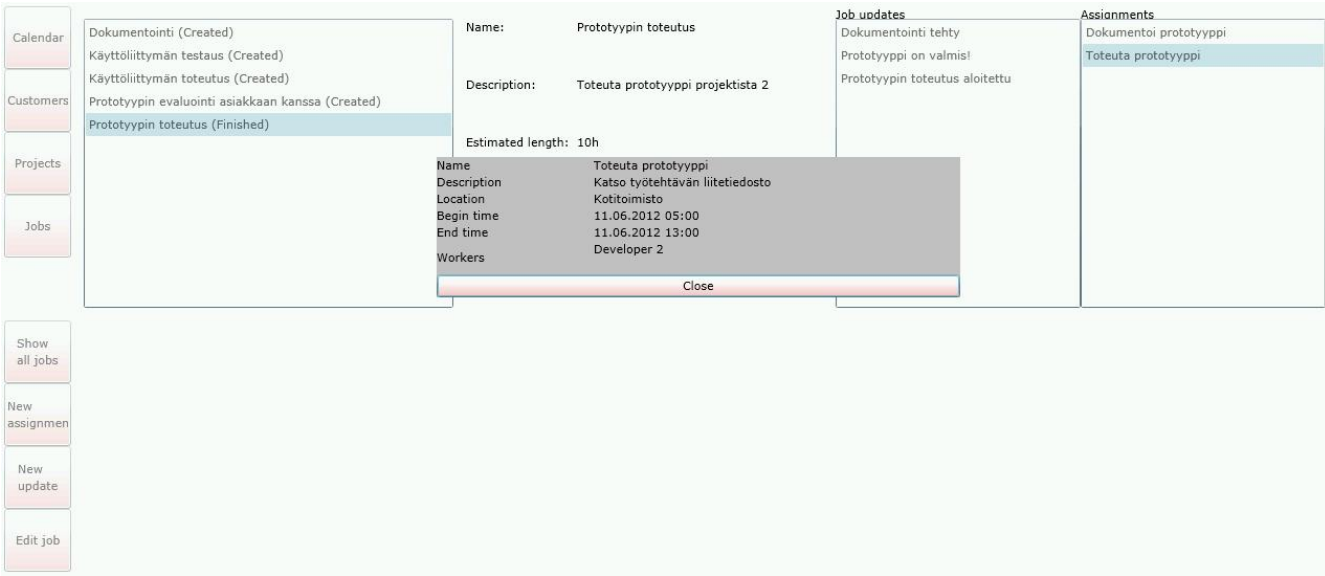
Kuva: Työjonon Silverlight-käyttöliittymän kalenterinäkymä. Kaksi työntekijää valittuna.



Kuva: Työjonon Silverlight-käyttöliittymän kalenterinäkymä. Viisi työntekijää valittuna, joista yksi tarkemmassa tarkastelussa.

(jatkuu)

LIITE 3. (jatkoa)



Kuva: Työjonon Silverlight-käyttöliittymän työnäkymä, jossa näkyvillä työtehtävän tarkemmat tiedot.

Kuva: Työtehtävän lisääminen Työjonon Silverlight-käyttöliittymässä.

(jatkuu)

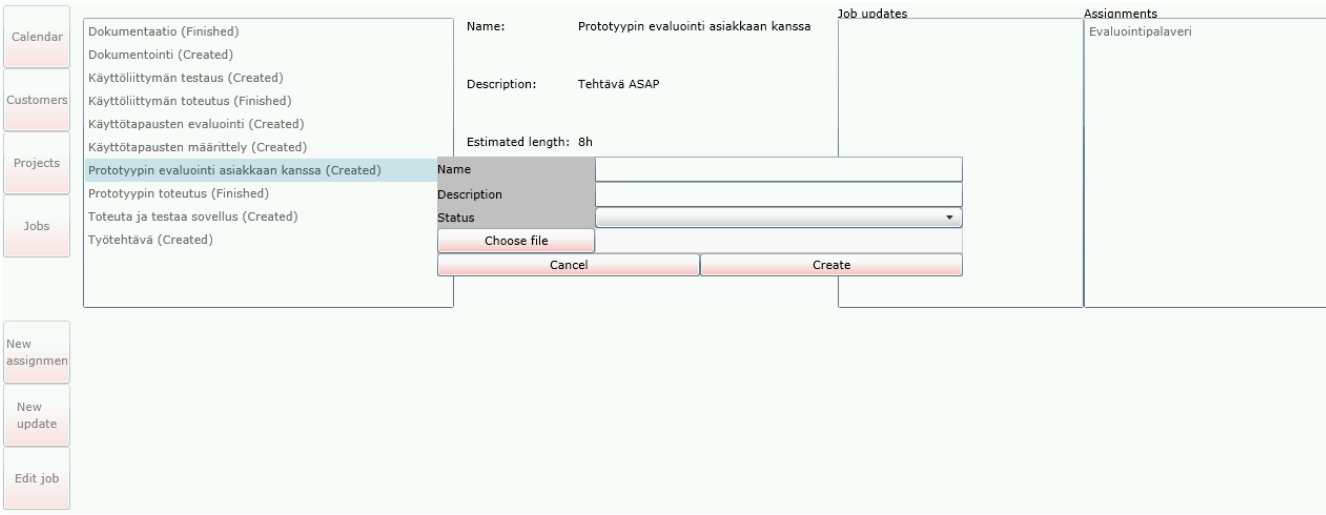
LIITE 3. (jatkoa)

Kuva: Työjonon Silverlight-käyttöliittymän kalenterinäkymä, jossa korostettuna valitun projektin ajanvaraukset.

(jatkuu)

LIITE 3. (jatkoa)

Kuva: Työjonon Silverlight-käyttöliittyän kalenterinäkymä, jossa korostettuna valitun työtehtävän ajanvaraus.



Kuva: Projektipäivityksen lisääminen Työjonon Silverlight-käyttöliittymässä.