AJANVARAUSJÄRJESTELMÄN VAATIMUSMÄÄRITTELY

Tuukka Tiainen

Opinnäytetyö
Helmikuu 2012

Tietojenkäsittely
Luonnontieteiden ala

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖN
KUVAILULEHTI

Tekijä(t) TIAINEN, Tuukka	Julkaisun laji Opinnäytetyö	Päivämäärä 06.02.2012
	Sivumäärä 40	Julkaisun kieli Suomi
	Luottamuksellisuus ( ) saakka	Verkkojulkaisulupa myönnetty (X)
Työn nimi AJANVARAUSJÄRJESTELMÄN VAATIMUSMÄÄRITTELY		
Koulutusohjelma Tietojenkäsittelyn koulutusohjelma		
Työn ohjaaja(t) TUIKKA, Tommi		
Toimeksiantaja(t) Oktacode osk		

Tiivistelmä

Opinnäytetyössä tehtiin vaatimusmäärittely tuotepohjaiselle ajanvarausjärjestelmälle. Lisäksi työssä kehittyi toimeksiantajalle vaatimustenhallinnan prosessi, jota voidaan käyttää jatkossa muiden tuotteiden elinkaaren aikana.

Opinnäytetyön toimeksiantaja on Oktacode osk, joka on jyväskyläläinen vuonna 2011 perustettu webohjelmointialan yritys. Oktacode on aloittamassa ensimmäisen tuotteen kehitystä, jota varten se tarvitsee kattavasti tehdyn vaatimusmäärittelyn.

Työssä haettiin vahva teoreettinen pohja vaatimustenhallinnalle, jota sovellettiin case-tyyppisesti ajanvarausjärjestelmän määrittelyssä.

Opinnäytetyössä käytettiin tutkimusmenetelminä tapaustutkimusta ja käsitteellisteoreettista tutkimusta. Aihealueesta etsittiin ja tutkittiin ensin vahva teoreettinen pohja, jonka perusteella johdettiin tapauskohtainen vaatimusmäärittely.

Tuloksina työstä saatiin kattava järjestelmän vaatimusmäärittelyn dokumentti. Prosessikuvaus oli myös yks työn tuloksista. Lisäksi löydettiin kyseisen järjestelmän sidosryhmät.

Avainsanat (asiasanat)

vaatimustenhallinta, vaatimusmäärittely, prosessi, ajanvarausjärjestelmä

Muut tiedot

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION

Author(s)	Type of publication	Date
TIAINEN, Tuukka	Bachelor's Thesis	06.02.2012
	Pages 40	Language Finnish
	Confidential ( ) saakka	Permission for web publication (X)

Title
REQUIREMENTS SPECIFICATION OF APPOINTMENT SYSTEM

Degree Programme
Business Information Systems

Tutor(s)
TUIKKA, Tommi

Assigned by
Oktacode osk

Abstract

This thesis presents the requirements specification of an appointment system. In addition, a requirements engineering process for client is developed in this thesis. This process can then be used in the future with other possible products.

The thesis is assigned by Oktacode osk which is a young company founded in 2011. Oktacode operates in the field of web programming. They are about to begin developing their first product that requires a comprehensive requirement specification.

In the thesis there is an inclusive amount of theory collected and applied to requirement specification with a case-study approach.

The case study and the conceptual theoretical method are the research methods that were used in the thesis. At first a strong theoretical basis was searched and studied and then the requirement specification was derived from it.

As a result of the thesis a comprehensive requirement document was developed with a process description as one of the further results. Additionally, all the stakeholders of the system were found.

Keywords

requirements engineering, requirement specification, process, appointment system

Miscellaneous

SISÄLTÖ

KUVIOT

TAULUKOT

1 JOHDANTO

Vaatimustenhallinta on erittäin tärkeä osa ohjelmistotuotantoa. Se saatetaan joskus unohtaa tai aliarvioida kohtalokkain seurauksin. Vaatimustenhallinnan laiminlyöntiä pidetään useasti myös osasyynä projektien venymiseen tai siihen, että lopputuotteeseen ei olla tyytyväisiä. Aihealueesta löytyy paljon teoriaa ja teoksia, mutta tärkeämpää on sen soveltaminen käytäntöön ja eri organisaatioihin. Prosessi on lähes aina organisaatiokohtainen ja uniikki.

Toimeksiantajan opinnäytetyössä on Oktacode osk, joka on keväällä 2011 perustettu web-ohjelmointialan osuuskuntamuotoinen yritys, jossa itse olen osakkaana. Suurin osa yrityksen projekteista on ollut asiakaslähtöisiä internetsivuja. Nyt toimeksiantaja havittelee myös oman tuotteen rakentamista.

Oktacoden tarkoitus on saada markkinoille ajanvarausjärjestelmä, joka päihittää kilpailevat tuotteet. Tätä varten toimeksiantaja haluaa vaatimusmäärittelyn, jonka toteutan osana opinnäytetyötä. Toisena osana syntyy myös prosessikuvaus, jota toimeksiantaja voi jatkossa noudattaa rakentaessaan uusia tuotteita tai asiakaslähtöisiä järjestelmiä. Vaatimustenhallinnan prosessi sekä vaatimusmäärittely ovat erilaisia, sillä kyseessä on tuote. Painoarvoa laitetaan asiakaslähtöisyydelle ja tuotteen sidosryhmien tunnistamiselle.

Vaatimustenhallinta on osa-alue, joka tukee ohjelmiston suunnittelua ja itse ohjelmointia. Tämä on tärkeä alue hallita myös työelämän kannalta. Yritysmaailmassa on ihmisiä, joiden toimenkuva saattaa olla melkein pelkästään vaatimustenhallinta ja erityisesti vaatimusmäärittelyn tekeminen. Requirements engineer toiminimikkeen alla tehdään tärkeää työtä, ja ilman sitä on hyvin arveluttavaa ruveta rakentamaan minkäänlaista järjestelmää. Uskon, että erityisesti Suomessa tämän osa-alueen arvostus kasvaa merkitsevästi ja sen ammattilaisia tarvitaan jatkuvasti lisää.

2 TUTKIMUKSEN TAUSTA

Tässä osiossa käydään läpi tutkimuksen taustaa, tutkimusongelmia ja itse tutkimuksen toteutusta.

2.1 Tutkimusongelma

Opinnäytetyön tutkimusongelma on seuraavanlainen: *Miten ajanvarausjärjestelmälle tehdään vaatimusmäärittely?* Tähän ongelmaan ratkaisua yritetään hakea tutkimuskysymyksillä.

1) *Mitkä ovat ajanvarausjärjestelmän vaatimukset?*
    I. *Mitkä ovat käyttäjävaatimukset?*
    II. *Mitkä ovat toimeksiantajan vaatimukset ja tavoitteet?*
2) *Minkälainen vaatimusmäärittelyn prosessi on sopiva ajanvarausjärjestelmässä?*
3) *Mitkä ovat ajanvarausjärjestelmän sidosryhmät?*

2.2 Tutkimuksen toteutus

Tutkimus on osaksi tyypiltään käsitteellis-teoreettinen, jossa tutkija voi luoda mallin tai teorian, jolla kuvataan reaalimaailman jotakin ilmiötä. Malli voidaan johtaa deduktiivisesti tai induktiivisesti eli joko reaalimaailman osaa koskevista olettamuksista tai sitten käyttämällä aikaisempia empiirisiä tutkimuksia. (Järvinen & Järvinen 2000, 15.)

Toisaalta Järvinen ja Järvinen kirjoittavat, että tietosysteemin rakentaminen on niin sanottu supermetodi, jolle muut metodit ovat alisteisia. Tämä johtuu siitä, että tietosysteemin rakentamisessa käytetään useita eri metodisia lähestymistapoja. (Järvinen ym. 2000, 102.)

Vaikka kyseessä ei ole kokonaisen tietojärjestelmän rakentaminen, voidaan muun muassa vaatimusmäärittelyä edeltävää jo valmiiden teorioiden tutkimista pitää käsitteellis-teoreettisena. Tutkimuksen onnistuessa yritys saa itselleen sopivan vaatimustenhallinnan prosessin, joka voidaan rinnastaa käsitteellis-teoreettiseen tuotokseen, jossa vanhoista teorioista johtamalla on saatu tilanteeseen sopivampi versio.

Työssä sovelletaan myös case- eli tapaustutkimusta. Tämä on empiirinen tutkimus, jossa voidaan käyttää eri tavoilla hankittua tietoa, jotta voidaan analysoida tutkittua tapahtumaa sen rajatussa ympäristössä. Case- eli tapaustutkimuksessa määritellään ensin tutkimuksen tavoitteet ja sen kohde. Seuraava askel on aineiston kokoaminen, minkä jälkeen informaatio järjestetään selvään ja kiinteään muotoon, joka kuvaa tutkimuskohdetta. Tämän jälkeen raportoidaan tutkimustulokset ja tarkastellaan niitä. Casetutkimus sopii mainiosti juuri prosessien tutkintaan. (Case-tutkimus n.d.)

3 VAATIMUSTENHALLINTA

Vaatimukset ovat projektin pohja. Vaatimuksenhallinta ei ole pelkästään asioiden kirjaamista, joita asiakas kertoo haluavansa. Vaatimusten saaminen asiakkaalta ei aina myöskään ole helppoa. Organisoitu vaatimustenhallinta on tarpeellinen tekijä huolimatta projektin koosta. (Young 2004, 2–3.)

Termi vaatimustenhallinta tulee englannin kielen sanasta requirements engineering. Termi on suhteellisen uusi. Sen on tarkoitus sisältää kaikki toimenpiteet, jotka liittyvät tietokonejärjestelmän vaatimusten löytämiseen, dokumentointiin ja ylläpitämiseen. Vaatimustenhallinnalla on paljon yhteneväisyyksiä järjestelmäanalyysin kanssa. Analyysin kuuluisi pääasiallisesti tutkia järjestelmän liiketoiminnallista puolta, mutta kuten vaatimustenhallintakin, se ottaa huomioon molemmat puolet, sekä järjestelmän että liiketoiminnan. (Kotonya & Sommerville 1998, 8.)

Vaatimustenhallinta on mukana projektin tai tuotteen koko elinkaaren ajan. Huomionarvoinen asia on myös vaatimustenhallinnan riippumattomuus alasta. Sitä voidaan käyttää yhtälailla laitteiston kuin sähkömekaanisen järjestelmänkin kehityksessä. (Berenbach, Paulish, Kazmeier & Rudorfer 2009, 2.)

Vaatimustenhallinta pitää sisällään paljon eri vaiheita ja asioita. Se sisältää kommunikoinnin kehitysryhmän, asiakkaan, käyttäjien ja muiden välillä. Toiminnallisten tarpeiden kuvailu ja ymmärrys on yksi osa vaatimustenhallintaa. Ratkaisut teknisistä päätöksistä kuuluvat myös vaatimustenhallintaan. Suuri osa itse tuotteen kehitystä on tuotteen arkkitehtuurin määrittely, joka on tärkeä osa prosessia. Tuotteen tai projektin elinkaaren kehitysvaiheen loppupäästä löytyvät tuotteen valmiuden toteaminen ja tuotteen valmiuden vahvistaminen. (Young 2004, 15.)

Varsinaista suomenkielistä käännöstä englanninkieliselle requirements engineering sanalle ei löydy. Muutenkin requirements engineering sanaa on käytetty hieman eri merkityksissä riippuen lähteestä. Esimerkiksi Kotonya ja Sommerville (1998, 8) määrittelevät termin olevan joukko prosesseja, jotka liittyvät vaatimusten löytämiseen, dokumentointiin ja ylläpitämiseen. Toisaalta Berenbach ja muut (2009, 8) määrittelevät requirements engineeringin olevan mukana kaikissa projektin tai tuotteen elinkaaren vaiheissa innovaatiosta vanhentumiseen saakka.

Requirements engineering on joskus yhdistetty englannin kielen termeihin requirements gathering, requirements capture tai requirements specification. Näiden termien välillä on syytä tehdä selvä pesäero. Jälkimmäisenä mainitut kolme termiä on yhdistettävissä suomen kielen sanaan vaatimusmäärittely. Tässä työssä vaatimusmäärittely on vaatimustenhallinnan yksi osa-alue. Vaatimusmäärittelyssä etsitään ja tutkitaan vaatimuksia. Vaatimustenhallinta on siis tuotteen tai projektin kaikissa elinkaaren vaiheissa mukana.

3.1 Vaatimukset

Vaatimukset ovat selvitys järjestelmän palveluista tai rajoituksista. Ne määritellään alkuvaiheessa ennen järjestelmän kehitystä. Vaatimukset kertovat, miten järjestelmän tulisi käyttäytyä. Vaatimus voi kuvata käyttäjätason ominaisuutta, yleisen tason ominaisuutta, järjestelmän ominaisuutta tai järjestelmän kehitystä. (Kotonya ym. 1998, 6–7.)

Tyypit

Vaatimuksia on projekteissa aina paljon erilaisia. Ne myös vaihtelevat laadultaan ja ominaisuuksiltaan. Sen takia ne on jaoteltu erilaisiin tyyppeihin. Yritysmaailmassa tärkeitä vaatimuksia ovat liiketoiminnan vaatimukset. Nämä vaatimukset johdetaan liiketoiminnan tavoitteista. (Young 2004, 49–50.)

Erityisesti vaatimusmäärittelyn alkuvaiheessa, jossa vasta tunnistetaan vaatimuksia, tulee esille paljon pyyntöjä. Englannin kielessä tässä vaiheessa tulevat esiin sanat ”stated requirement” ja ”real requirement”. Ainoastaan jälkimmäisten kuuluu päästä mukaan valmiiseen järjestelmään. (Young 2004, 50.)

Toiminnallinen vaatimus tulee esille lähes aina, kun puhutaan vaatimusmäärittelystä. Se määrittelee, mitä järjestelmän tulee tehdä. Vaatimus kertoo sen, mitä järjestelmässä tehdään, että jotain haluttua tapahtuu. Toinen yleinen vaatimustyyppi on ei-toiminnallinen vaatimus. Nimi viittaakin siihen, että se ei ole itsessään järjestelmän toimintaan liittyvä vaatimus. Sen sijaan ei-toiminnallinen vaatimus määrittelee järjestelmän ominaisuuksia, kuten luotettavuus ja turvallisuus. (Young 2004, 51.)

Käyttäjävaatimukset ovat yksi vaatimustyyppi. Käyttäjät ovat yksilöitä tai ryhmiä, jotka käyttävät järjestelmää tai ohjelmaa sen ympäristössä. Käyttäjävaatimukset ovat edellä mainittujen käyttäjien todennettuja vaatimuksia, jotka koskevat järjestelmää tai ohjelmaa. Käyttäjälähtöinen vaatimus on myös liiketoiminnan sääntö, englanniksi ”business rule”. Järjestelmän tulee tukea yrityksen menettelytapoja, rajoituksia ja

liiketoimintaa. Muita vaatimuksia voivat olla esimerkiksi liiketoiminnalliset määritykset tai suhteet ja työnkulku. (Young 2004, 51.)

Hyvä vaatimus

Ei ole yksiselitteistä kuvata hyvää vaatimusta. Vaatimuksella on kuitenkin erilaisia ominaisuuksia. Jos ominaisuudet ovat oikeanlaiset, vaatimusta voidaan pitää hyvänä. Seuraavassa käydään läpi vaatimuksen hyviä ominaisuuksia.

Ensinnäkin vaatimuksen tulee olla toteutettavissa. Sitä se on, jos sen täytäntöönpano on mahdollista suunnitellussa ohjelmassa tai projektissa sen rajoitteiden puitteissa. Vaatimuksen toteutettavuus on tietenkin suhteellista, koska tietty vaatimus voi olla toteutettavissa toisella alustalla, mutta toisella alustalla sitä ei voida toteuttaa. Lyhyesti sanottuna vaatimus on toteuttavissa, jos se on mahdollista tehdä annetuilla resursseilla, budjetilla, taidoilla ja saatavilla olevilla tekniikoilla. (Berenbach ym. 2009, 9.)

Ohjelmistoalalla käytetään useasti suomalaistunutta termiä validi. Sana tulee tietenkin englannin kielen sanasta valid. Tässä työssä käytetään sen suomennosta pätevä. Vaatimus on pätevä ainoastaan, jos se kannattaa ottaa järjestelmään. Vaatimuksia selvitettäessä tulee esille erilaisia vaatimuksia. Jotkut näistä voivat olla vaatimuksia, jotka järjestelmässä pitää ehdottomasti olla, jotkut taas ovat niin sanottuja ”olisi kivaa saada” -vaatimuksia. Järjestelmään kannattaa luonnollisesti ottaa vain pätevät vaatimukset. Epäpätevät vaatimukset lisäävät järjestelmän hintaa tuomatta lisäarvoa ja mahdollisesti viivyttävät projektia. (Berenbach ym. 2009, 9.)

Yksiselitteisyys on tärkeä ominaisuus vaatimukselle. Vaatimus on yksiselitteinen, kun se voidaan tulkita vain yhdellä tavalla. Jos vaatimus ei ole yksiselitteinen, se saattaa jättää tulkinnanvaraa kehitysvaiheessa, mikä johtaa siihen, että ohjelmistokehittäjä tuottaa jopa vääränlaisen ominaisuuden. Yhtenä vaatimusmäärittelyn laadun mittavälineenä voidaan pitää epäselvien vaatimusten prosenttiosuutta vaatimuksista. (Berenbach ym. 2009, 11.)

Vaatimuksen on oltava todennettava, jotta se on hyvä vaatimus. Todennettavuus täyttyy, jos valmis järjestelmä tai tuote voidaan testata varmistukseksi siitä, että se vastaa haluttua. Vaatimuksista on valmiissa tuotteessa tullut ominaisuuksia. Kaikkia ominaisuuksia ei kuitenkaan voida välttämättä testata. (Berenbach ym. 2009, 11.) Järjestelmälle voidaan esimerkiksi laittaa vaatimus, että sen pitää käynnistyä nopeasti. Tämä ei ole todennettava vaatimus. Todennettava versio tästä vaatimuksesta voisi olla seuraavanlainen: järjestelmän pitää käynnistyä 10 sekunnissa.

3.2 Vaatimustenhallinnan tärkeys

Projektien onnistumisesta on tehty lukuisia tutkimuksia ja kyselyitä. Useat näistä todistavat, että projektit ovat epäonnistuneet, koska vaatimusprosesseja ei ole otettu huomioon tarpeeksi hyvin. Erään kyselyn mukaan, jonka on tehnyt The Standish and Gartner groups, 26 prosenttia järjestelmäprojekteista onnistuu ja 74 prosenttia epäonnistuu. Tutkittaessa onnistumista tai epäonnistumista nousevat esille useimmiten käyttäjävaatimukset. Toisen heidän julkaisemansa raportin mukaan melkein puolet projekteista epäonnistui tai lopetettiin puutteellisen vaatimustenhallinnan takia. (Aouad & Arayici 2010, 4.)

Ohjelmistoprojekteja on saatettu onnistuneesti loppuun jo vuosikymmenien ajan ilman vaatimustenhallintaan erikoistuneita ammattilaisia. Miksi vaatimustenhallinnan tärkeys on korostunut nykypäivänä? Vastaukset löytyvät tuotannon ja yhteisön muutoksista. Muutamia vuosikymmeniä sitten tuotteenkehitys oli hidasta. Nykyään asiakkaat saattavat vaatia tuotteesta uusia versioita vuosittain. Siemensin mukaan noin parikymmentä vuotta sitten 55 prosenttia myynnistä koostui tuotteista jotka olivat alle viisi vuotta vanhoja. Nykyään vastaava prosenttiluku on 75. (Berenbach ym. 2009, 2.)

Toinen vaikuttava tekijä muutoksessa on työpaikkojen pysyvyys. Joitakin vuosia sitten insinöörit ja muut ammattilaiset saattoivat olettaa tekevänsä töitä

samalla yritykselle koko uran ajan. Nykyään työpaikan vaihtuvuus on suurempaa. Yksi vaatimustenhallinnan tärkeyteen vaikuttava tekijä on ulkoistaminen sekä tuottamisen vieminen ulkomaille. Voidaan vain kuvitella, miten tärkeää vaatimusten rooli on tilanteessa, kun esimerkiksi pesukonetta rakentavat ihmiset, jotka eivät koskaan ole nähneet vastaavaa. (Berenbach ym. 2009, 2.)

Väärillä tai viallisilla vaatimuksilla voi olla erilaisia seuraamuksia. Useasti epäonnistunut vaatimusmäärittely saattaa aiheuttaa projektin tai tuotteen viivästymisen. Tämä yleensä myös lisää hintaa tuotteelle. Toinen mahdollinen seuraamus on asiakkaan ja loppukäyttäjien tyytymättömyys järjestelmään. On mahdollista, etteivät he käytä sen toimintoja tai jopa hylkäävät koko järjestelmän. Yksi epäonnistumisen ilmentyminen on epäluotettavuus. Järjestelmässä saattaa tulla säännöllisesti virheitä ja järjestelmä saattaa jopa kaatua, jolloin tavallinen käyttö vaikeutuu huomattavasti. (Kotonya ym. 1998, 8.)

Siinä tapauksessa, että tällaista järjestelmää halutaan silti käyttää, ylläpitämisen ja kehittämisen kulut ovat yleensä korkeat. Tämä johtuu yksinkertaisesti siitä, että järjestelmän alkuvaiheessa vian korjaaminen on monin kerroin helpompaa ja halvempaa. Vaatimuksista johtuvan ongelman korjaaminen saattaa aiheuttaa järjestelmän suunnittelun, implementaation ja testauksen uusimisen. On laskettu, että vaatimuksesta johtuvan vian korjaaminen on noin sata kertaa kalliimpaa kuin perinteisen ohjelmointivirheen korjaaminen. (Kotonya ym. 1998, 9.)

3.3 Vaatimustenhallinnan onnistumisen lähtökohdat

Vaikka vaatimustenhallinta ei ole mitenkään ylipääsemättömän vaikeaa, niin liian usein sitä pidetään oletusarvoltaan hankalana. Yritys tai organisaatio voi yksinkertaisilla asioilla edistää vaatimustenhallinnan onnistumista. Seuraavassa käsitellään onnistumiseen vaikuttavia tekijöitä.

Projektilla, jolla on pääarkkitehti työllistettynä, on suurempi mahdollisuus onnistua. Pääarkkitehti tuo projektiin teknistä jatkuvuutta ja näkökulmia. Arkkitehti on myös vastuussa ei-toiminnallisten vaatimusten hallinnoinnista. Näitä vaatimuksia voivat olla laatu, suorituskyky, ympäristö, skaalautuvuus ja niin edelleen. (Berenbach ym. 2009, 5.)

Vaatimusten tunnistaminen tulee alkaa jo myyntitiimin osalta. On hyvin tärkeää, että projektin vaatimustenhallinnan työntekijöillä on hyvä suhde myyntiorganisaation kanssa. Tämä helpottaa vaatimusten löytämistä, ja vaatimuksista tulee suuremmalla todennäköisyydellä pätevämpiä. Tunnetusti väärät vaatimukset tai ominaisuudet tuottavat enemmän ongelmia projektin edetessä. (Berenbach ym. 2009, 6.)

Vaatimuksia tulee tarkastella kriittisesti. Tämän vuoksi katselmustilaisuuksia on järjestettävä. Kun vaatimuksia on neljästä kymmeneen, niin tarkastelu vie aikaa noin tunnin. Tarkastelun ohella tärkeä seikka on vaatimussuunnittelijoiden kokemustaso. Ilman kokemusta projekti voi viivästyä ja sekoittua. Vaatimussuunnittelija oppii taitonsa harjoittelemalla. Taito ja kokemus kuitenkin ovat kaksi eri asiaa. Siksi suositellaankin, että projektilla olisi hyvä olla kokeneempi opastaja, jos suunnittelijat ovat uusia tai ryhmässä on yli 4 jäsentä. (Berenbach ym. 2009, 6.)

Toinen puoli projektista, eli asiakas, on syytä hoitaa asianmukaisesti. Tämä sisältää muun muassa nopeaa palautetta ja sen keräämistä mahdollisessa protoiluvaiheessa. Asiakashallinnan on erityisesti määrittelyvaiheessa oltava aktiivista, mutta myös tiukkaa. Yksi huomionarvoinen seikka vaatimustenhallinnan onnistumisessa ovat erilaiset työkalut. Vaatimustenhallintaan on kehitetty erilaisia sovelluksia, jotka helpottavat itse työtä. Näitä suositellaan käytettäväksi. (Berenbach ym. 2009, 7.)

3.4 Vaatimusmäärittely

Vaatimustenhallinnan yksi tärkeimmistä osa-alueista on vaatimusmäärittely. Se pitää sisällään vaatimusten analyysin, koostamisen, määrittelyn ja validoinnin. Vaatimustenhallinta on kiinnostunut jokapäiväisistä ongelmista, jotka liittyvät järjestelmään. Käyttäjien tehtävät ja toiminnot sekä liiketoiminnan menetelmät ovat yleensä monimutkaisia. Kun tähän vielä lisätään ihmisiä organisaation eri tasoilta, tilanne muuttuu entistä monimuotoisemmaksi. Tästä johtuen vaatimukset ovat useasti monimutkaisia. (Aouad ym. 2010, 4.)

Vaatimustenhallinta on iteratiivinen prosessi. Ryhmien ja yksittäisten henkilöiden tarpeet, asetukset ja vaatimukset on tutkittava, tunnistettava, kartoitettava ja mallinnettava. Asiakkaan, käyttäjän ja markkinoiden vaatimukset on myös tunnistettava. Toisaalta yhtä tärkeää on määritellä suunnittelun vaatimukset ja tekniset vaatimukset. (Aouad ym. 2010, 14.)

Vaatimusdokumentti on selostus vaatimuksista ohjelmistokehittäjille, asiakkaille ja loppukäyttäjille. Dokumentille on useita eri nimityksiä. Englannin kielessä näitä ovat esimerkiksi ”functional specification”, ”the requirements defition” tai ”the software requirements specification”. (Kotonya ym. 1998, 9.) Tässä tutkimuksessa dokumentista käytetään termiä vaatimusmäärittelyn dokumentti tai vaatimusdokumentti.

Organisaation pitää ymmärtää, mitä tilaaja haluaa. Sen takia tehdään vaatimusmäärittely ennen itse suunnittelua tai toteutusta. Määrittely toimii myös kaksisuuntaisena vakuutena, että osapuolet ymmärtävät toisiaan. Itse vaatimusmäärittelydokumentti toteaa toiminnallisuudet ja valmiudet, jota järjestelmän tulee tarjota. Toisaalta dokumentista selviävät myös järjestelmän rajoitukset. (Le Vier 2010.)

Sidosryhmät (stakeholders)

Järjestelmän stakeholder eli sidosryhmän jäsen on joko ihminen tai organisaatio, johon järjestelmä vaikuttaa. Sidosryhmillä on suora tai epäsuora vaikutus järjestelmän vaatimuksiin. Kyseinen joukko voi pitää sisällään loppukäyttäjiä, kehityksestä ja ylläpidosta vastaavia insinöörejä tai organisaation asiakkaita. (Kotonya ym. 1998, 10.) Yhtenä onnistumisen avaimena vaatimustenhallinnassa on stakeholderien huomioon ottaminen. Jos vaatimukset halutaan määritellä ja priorisoida hyvin, niin kaikki oleelliset sidosryhmät on tunnistettava. (Berenbach ym. 2009, 7.)

Vaatimusmäärittelyn ongelmat

Liian usein vaatimusten keräämiseen asetetaan yrityksen uudet työntekijät. On jopa mahdollista, että vaatimukset kerätään vasta siinä vaiheessa, kun sovellus on lähes valmis ja tarvitaan vaatimukset testitapauksia varten. Tilanne voi olla vieläkin huonompi, jolloin tajutaan, että vaatimuskatselmus on pakollinen osa hyväksyntää tai jopa maksua varten. Vaatimusten löytäminen saattaa olla hieman ongelmallista lähes valmiista sovelluksesta. (Berenbach ym. 2009, 41.)

Kokemuskaan ei välttämättä aina riitä vaatimusmäärittelyssä. Vaatimusanalyytikot eivät välttämättä paneudu tai pysty paneutumaan alakohtaiseen taustatietoon ennen tapaamista sidosryhmien kanssa. Tästä voi mahdollisesti seurata turhautumista sidosryhmissä tai jopa väärinkäsityksiä vaatimuksissa, koska käytetään alakohtaista erikoistermistöä. (Kotonya ym. 1998, 62.)

Kyseenalaistaminen on tärkeä asia vaatimusten kartoituksessa. Vaatimuksia kerättäessä on elintärkeää, että joku kyseenalaistaa vaatimuksia. Ryhmän tai henkilön, joka vaatimuksia kerää, on tunnistettava se oikea tarve vaatimukselle. Muutoin valmiiseen tuotteeseen asti tiensä saattaa löytää täysin turhia vaatimuksia, jotka ovat vain roikkuneet mukana. (Berenbach ym. 2009, 41.)

Yksi yleinen ongelma vaatimuskartoituksessa on liiallinen vaatimusten kerääjän tekninen orientoituminen. Hän saattaa ajatella vaatimuksia liian teknisesti siinä vaiheessa, kun ollaan vasta hakemassa niitä sidosryhmiltä. Tämä saattaa johtaa vaatimusten vääristymiseen. (Berenbach ym. 2009, 42.) Toisaalta tekninen orientoituminen ei itsessään ole pahasta, mutta jos se on ainut perspektiivi, josta asiaa osataan katsoa, sitä voidaan pitää ongelmana.

Erityisesti suuremmissa projekteissa, joissa on paljon eri osapuolia mukana, voi olla vaikeaa tunnistaa tarkasti eri sidosryhmät. On hyvinkin mahdollista, että sidosryhmien palaverissa on mukana esimerkiksi 10–15 eri osapuolta. Jokaisella on omat vaatimuksensa ja keskustelu voi olla hektistä. Tällöin sidosryhmien tietojen tallentaminen on erityisen tärkeätä. (Berenbach ym. 2009, 44.)

Sidosryhmät eivät aina ole yhteistyöhaluisia. Uudet järjestelmät tai ohjelmat ovat lähes aina organisaatiotason päätöksiä eikä niistä välttämättä keskustella sidosryhmien kanssa. Tämä saattaa herättää sidosryhmissä turhautuneisuutta, joka mahdollisesti vaikeuttaa vaatimusanalyytikon työtä keräämisvaiheessa. Yksi ongelma vaatimusten kartoituksessa sidosryhmien osalta on ajankäyttö. Heillä ei ole välttämättä tarpeeksi aikaa omien töidensä ohella paneutua uuteen järjestelmään vaatimusanalyytikon kanssa. (Kotonya ym. 1998, 62.)

Joskus erinäisten syiden takia vaatimusten muuttuminen saattaa olla melko suurta. Tähän vaikuttaa taas tarpeiden muuttuminen tai se, että asiakas ei osaa päättää tai tehdä ratkaisua. On mahdollista, että tarpeiden vaihtuessa nopeasti vaatimukset eivät ole käyttökelpoisia. Tästä johtuen saatetaan joutua odottamaan, kunnes saavutetaan tietty vakaus. Vasta sen jälkeen yritetään valmistella vaatimukset toteutettaviksi. (Berenbach ym. 2009, 45.)

Useasti ongelmat vaatimusten kartoituksessa ovat asiakaslähtöisiä. Yksi suuri ongelma saattaa olla asiakkaan ymmärtämättömyys omista tarpeista tai jopa liiketoiminnan tavoitteista. Silloin vaatimusmäärittelijältä tarvitaan ammattitaitoa, jotta saadaan asiakkaalta toteutettavat vaatimukset.

Asiakkaalla ei välttämättä ole samanlaista tietoa tekniikasta ja tietokoneista, jota tietojärjestelmiä tai ohjelmia tuottavalla yrityksellä on. Tästä johtuen asiakas saattaa tuottaa futuristisia, toiveajatuksellisia tai epäkäytännöllisiä vaatimuksia. (Berenbach ym. 2009, 46–47.) Asiakas saattaa itse ymmärtää, mitä hän haluaa, mutta sen ulostuottaminen teknisessä muodossa saattaa olla vaikeaa. Asiakkaan sidosryhmät voivat vaatia myös mahdottomia, koska ne eivät tiedä kustannuksia. (Kotonya ym. 1998, 56.)

Hyvänä esimerkkinä asiakkaan teknisestä tietämättömyydestä on vuonna 1992 tehdyssä vaatimusmäärittelyssä. Lausunnossa todetaan seuraavaa: koska tietokoneissa ei tule ikinä olemaan tarvetta useammalle kuin yhdelle prosessorille, ei ole tarvetta tehdä myöskään järjestelmää tukemaan useampia prosessoreita. (Berenbach ym. 2009, 47.)

3.5 Vaatimusten kartoitus (requirement elicitation)

Järjestelmän vaatimukset saadaan selville neuvoteltaessa sidosryhmien kanssa, järjestelmän dokumenteista, alan tuntemuksella ja markkinatutkimuksilla. Englannin kielessä tälle prosessin osalle synonyymeja ovat ”requirements acquisition” tai ”requirements discovery”. (Kotonya ym. 1998, 32.) Jos vaatimusten keräys ei ole tehokasta ja siihen ei panosteta, ongelmat kasvavat entisestään projektin edetessä. (Young 2004, 2–3.)

Vaatimusten kartoituksessa on tärkeää paneutua alaan, johon järjestelmää tai ohjelmaa ollaan toteuttamassa. Esimerkiksi, jos toteutetaan vaatimuksia rautatien signaalijärjestelmään, on suunnittelijoilta löydyttävä taustatietoa rautateiden toiminnasta ja junien fysiikasta. (Kotonya ym. 1998, 55.)

Jotta asiaan voidaan päästä syvemmälle, on ymmärrettävä yksityiskohdat asiakkaan ongelmasta jota varten järjestelmää ollaan tekemässä. Rautatie-esimerkkinä suunnittelijan on tiedettävä nopeusrajoitukset rautatien eri

kohdissa signaalijärjestelmää varten. Ongelman ymmärtämisen aikana yleisen alakohtaisen tietämyksen tulee kasvaa. (Kotonya ym. 1998, 55.)

Asiakaslähtöinen ohjelmistokehitys hakee yleensä ratkaisua tiettyyn ongelmaan.

Usein tämä ongelma löytyy asiakkaan liiketoiminnasta. Järjestelmän tarkoitus on osallistua liiketoiminnan tai organisaation kehitykseen. Suunnittelijalla tulee olla ymmärrys siitä, miten mikäkin järjestelmän osa toimii liiketoimintaa ajatellen ja miten järjestelmä kokonaisuutena osallistuu liiketoiminnan tavoitteiden saavuttamiseen. (Kotonya ym. 1998, 55.)

Edellä mainittujen seikkojen ohella sidosryhmät ovat vaatimusmäärittelyssä tärkeässä osassa. Määrittelyssä tulee ymmärtää järjestelmän sidosryhmien tarpeet ja rajoitukset. Järjestelmän tulee tukea heidän työtään. Sen takia pitää ymmärtää, miten sidosryhmät käyttävät järjestelmää ja mitä ne siltä vaativat. Toinen yhtä tärkeä asia on ymmärtää työprosessit ja se, miten mahdollinen vanha järjestelmä niitä tukee. (Kotonya ym. 1998, 55.)

Berenbach (2009, 69) kirjoittaa, että on olemassa paljon erilaisia tekniikoita vaatimuskartoitukseen, mutta vaatimusanalyytikoiden on aina saatava koulutus kyseiseen metodiin. Hänen mielestään metodien ja keräysmekanismien on oltava hyvin määriteltyjä ja dokumentoituja. Nämä metodit ja tekniikat on hyvä kertoa sidosryhmille jo ennen vaatimuskartoitustapaamisia.

3.6 Vaatimusten analyysi ja neuvottelut

Vaatimusten kartoitus sekä analyysi ja neuvottelut eivät vaiheina selvästi jaksotu peräkkäin toisiinsa nähden. Monilla on harhaluulo, että kartoituksen loppuessa analyysivaihe alkaa. Todellisuudessa analyysi ja kartoitus ovat limittäin keskenään. Analyysi ja neuvottelut voivat alkaa, kun kartoitus on

saanut tarpeeksi vaatimuksia selville. Nämä vaiheet voivat mahdollisesti tapahtua syklisesti projektin aikana, sillä ongelmia saattaa ilmetä tai asioita on saattaa jäädä huomaamatta. Silloin ne toistuvat yhä uudelleen, niin kauan kunnes vaatimuksiin ollaan tyytyväisiä. (Kotonya ym. 1998, 52.)

Vaatimukset tulee analysoida yksityiskohtaisesti ja neuvotteluita tulee käydä eri sidosryhmien kanssa, jotta saadaan selvyys, mitkä vaatimukset hyväksytään. On useita syitä, miksi tämä prosessin osa on tärkeä. Vaatimukset saattavat olla erilaisia lähteestä riippuen. On myös mahdollista, että tieto saattaa olla vaillinaista tai esille tulleet vaatimukset eivät välttämättä sovi budjettiin, joka on määritelty järjestelmän kehitykseen. Yleensä kuitenkin vaatimuksista löytyy joustoa. Analyysi ja neuvottelut ovat siis tärkeä osa prosessia, jotta voidaan päättää oikeat vaatimukset järjestelmälle. (Kotonya ym. 1998, 32.)

Analyysivaiheessa vaatimusanalyytikot viettävät aikansa tutkimalla vaatimusdokumentteja. Sitä pidetäänkin kalliina ja aikaa vievänä prosessina, koska projekti ei etene niin näkyvästi tässä vaiheessa. Analyysissa käytetään useasti tarkistuslistaa, koska ne muistuttavat siitä, mitä dokumentista kannattaa etsiä. Tarkastuslista hakee vastuksia muun muassa siihen, onko vaatimus varmasti yksittäinen vaatimus vai voiko sen jakaa useammaksi vaatimukseksi. Vaatimuksesta pitää myös tarkastaa sen tarpeellisuus. Joskus vaatimus on lähinnä kosmeettinen lisä ja ei tuo todellista lisäarvoa järjestelmälle. Yksi tarkastettava asia on yhteneväisyys liiketoiminnan tavoitteiden kanssa. Pääasiallisesti vaatimusten on tarkoitus tukea yrityksen liiketoimintaa, jolle järjestelmää tehdään. Hyvän vaatimuksen ominaisuudet on jo kerrottu aiemmin, mutta analyysissa on syytä tutkia, onko vaatimus yksiselitteinen. Toinen yhtä tärkeä asia on tarkastaa, voidaanko vaatimus toteuttaa olemassa olevalla teknologialla. Viimeisenä tulee selvittää, voidaanko vaatimus testata. (Kotonya ym. 1998, 79.)

Analyysin jälkeen tulee neuvotteluvaihe, jonka tarkoitus on ratkaista konfliktit, laiminlyönnit ja päällekkäisyydet vaatimuksista. Se sisältää edellä mainittujen asioiden ohella myös tietojen vaihtoa ja keskustelua, sekä tuottaa ratkaisuja

eriäviin mielipiteisiin. Kaikkia sidosryhmiä kaikki vaatimukset eivät voi tyydyttää. Sen takia neuvottelussa on löydettävä jokin kompromissiratkaisu, jonka kanssa kaikki sidosryhmät voivat elää. (Kotonya ym. 1998, 81.)

3.7 Vaatimusten dokumentointi

Seuraava askel on dokumentoida sovitut vaatimukset. Dokumentoinnin tulee tapahtua sopivan yksityiskohtaisella tasolla. Hyvä perussääntö on se, että kaikki sidosryhmät ymmärtävät dokumentoinnin. Käytännössä dokumentoinnin pitää tapahtua luonnollisella kielellä ja kaavioilla. (Kotonya ym. 1998, 33.)

Kotonya ja muut (1998, 14) kirjoittavat, että yhtä ainutta vakiintunutta termiä ei vaatimusdokumentille ole. Sillä on englannin kielessä useita eri nimiä, kuten ”requirements document”, ”the functional specification” ja ”the system requirements specification”.

Dokumentti sisältää tärkeää tietoa asiakkaalle, järjestelmän rakentajille ja projektin vetäjille. Sen tulee kertoa palvelut ja toiminnot, joita järjestelmän tulee toimittaa. Vaatimusdokumentista käyvät ilmi myös rajoitukset joiden lomassa järjestelmän tulee toimia. Yleisien ominaisuuksien ja erityisen tärkeiden ominaisuuksien rajoituksien tulee olla vaatimusdokumentissa. Mikäli järjestelmän tulee sopeutua toisten järjestelmien kanssa yhteen, on siitäkin löydyttävä määritelmä dokumentista. Hieman järjestelmästä itsestään poiketen, dokumentti kertoo mahdollisista rajoituksista itse järjestelmän rakennusvaiheessa. (Kotonya ym. 1998, 15.)

3.8 Vaatimusten validointi

Ennen järjestelmän kehitystä vaatimukset on tutkittava vielä tarkasti. Tämä prosessin osa tarkastaa vaatimusten johdonmukaisuuden ja täydellisyyden. Tärkein idea tässä vaiheessa on löytää mahdolliset ongelmat vaatimusdokumenteissa. (Kotonya ym. 1998, 33). Aouad ja muut (2010, 19) puolestaan kirjoittavat, että yksi vaatimusten validoinnin tarkoitus on varmistaa järjestelmäinsinöörien ymmärrys vaatimuksista.

Tässä vaiheessa on mahdollista tutkia, onko vaatimusdokumentti yleisesti yrityksen standardien mukainen. Validointiin on useita eri menetelmiä ja tapoja. Näitä voi esimerkiksi olla: dokumentin katselmus sekä läpikäynti -sessiot, prototyping tai testaus. (Aouad ym. 2010, 19.) Validaation läpäisseet vaatimukset lisätään toimitettavaan järjestelmään.

3.9 Vaatimusten hallinnointi

Tässä vaiheessa tulee tehdä selväksi ero itse vaatimustenhallinnan ja vaatimusten hallinnoinnin välillä. Vaatimusten hallinnointi tulee englannin kielen sanoista requirements management. Vaatimusten hallinnointi on siis osa vaatimusmäärittelyä, joka taas on osa kokonaisuutta eli vaatimustenhallintaa.

Vaatimusten hallinnointi toimii jokaisen edellä mainitun prosessin kanssa rinnakkain. Sen tarkoitus on pitää huolta mahdollisista vaatimusten vaihtumisista tai muutoksissa niistä. Muutokset vaatimuksissa ovat väistämättömiä liiketoiminnan prioriteettien vaihtuessa, virheiden tai pois jättämisten takia vaatimuksissa tai jos uusia vaatimuksia ilmaantuu. Vaatimusten hallinnoinnin tehtävä on pysyä ajan tasalla kaikista muutoksista ja tehdä muutokset dokumentteihin hallitulla tavalla. (Kotonya ym. 1998, 33.)

4 VAATIMUSTENHALLINNAN PROSESSI

Young (2004, 222) määrittelee vaatimustenhallinnan prosessin olevan joukon toimenpiteitä, jotka liittyvät järjestelmän tarpeellisiin ominaisuuksiin sen koko elinkaaren ajan. Prosessiin kuuluu asiakkaan tarpeiden ja odotusten ymmärtäminen. Edellä mainittua osaa kutsutaan myös vaatimusten kartoitukseksi. Prosessiin kuuluvat myös vaatimusten analysointi ja eritteleminen, priorisointi, vaatimusten johtaminen, jaottelu, osiointi, jäljittäminen, hallinnointi, todentaminen ja validointi.

Ihanteellista vaatimustenhallinnan menetelmää, joka sopisi kaikille organisaatioille, ei ole olemassa. Organisaation on kehitettävä sopiva prosessi eli menetelmä sen perusteella, millaista järjestelmää kehitetään. Prosessiin luonnollisesti vaikuttavat organisaatiokulttuuri sekä vaatimustenhallinnassa mukana olevien ihmisten kokemus ja kyvyt. Yleensä vaatimustenhallinnan järjestämiseen löytyy useita tapoja. Jotta voidaan määritellä organisaatiolle sopiva prosessi, pitää mukana olla ihmisiä, jotka ovat mukana myös vaatimustenhallinnassa. Joskus ulkopuolisen avun hankkiminen saattaa parantaa menetelmän kehitystä, koska se tuo lisää objektiivista näkökantaa. (Kotonya ym. 1998, 9.)

5 CASE: AJANVARAUSJÄRJESTELMÄ - AJAVA

Ajanvarausjärjestelmän vaatimusmäärittely on hieman tavallisesta ohjelmistoprojektista poikkeavaa. Ajava tulee olemaan Oktacode osk:n yksi tuote, jota se myy ajanvarausjärjestelmää tarvitseville asiakkaille. Esimerkiksi tavallisessa web-ohjelmointiprojektissa tehdään asiakkaalle kotisivut, jolloin vaatimukset saadaan kohtaamaan tarpeet suhteellisen helposti. Ajavan tapauksessa ei voida räätälöidä kaikille asiakkaille täysin uniikkia

ajanvarausjärjestelmää, vaan esimerkiksi käyttöliittymän on oltava mahdollisimman universaali.

Oletetusti myös sidosryhmien määrä on korkeampi kuin tavallisessa projektissa, koska kyseessä on tuote, jonka asiakkaille on vielä omat asiakkaat. Tuotteella on myös myyjät ja muut erityiset sidosryhmät verrattuna yksittäiseen projektiin. Ajavan vaatimusmäärittelyssä pyritään noudattamaan aiemmin tutkimuksessa esille tulleita käytäntöjä ja tapoja.

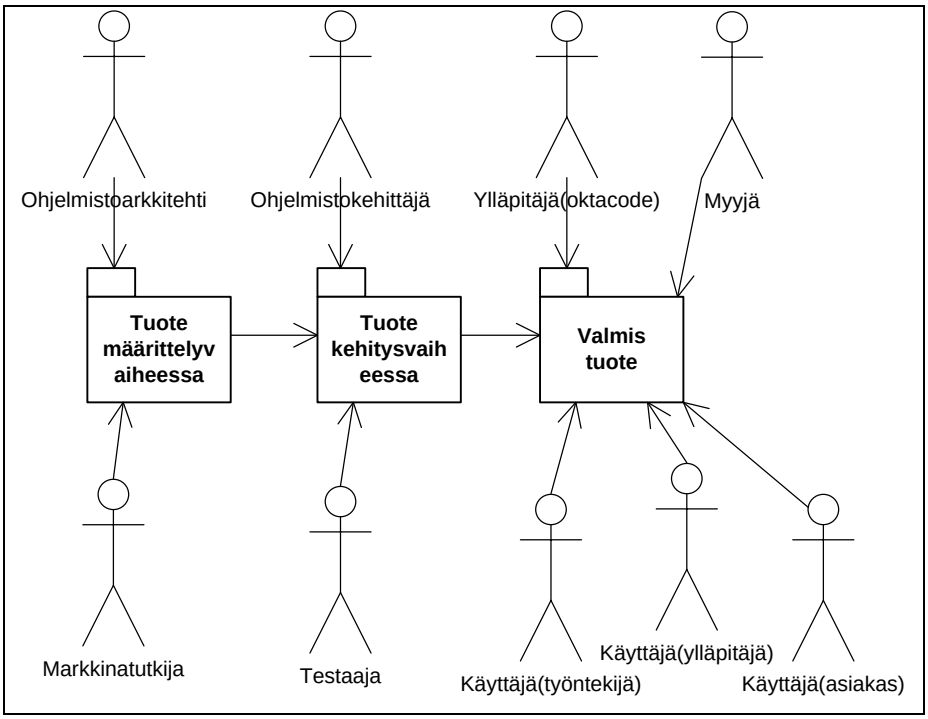
5.1 Liiketoiminnan tavoitteet ja ongelmat

Ajanvarausjärjestelmä on ohjelmisto, joka tullaan kehittämään tuotteeksi yrityksille. Tämän takia pitää ymmärtää, mitä liiketoiminnan tavoitteita ajanvarausjärjestelmä mahdollisilla asiakkailla edistää tai mitä ongelmia kohdeasiakkailla voi olla. Tästä johtuen Oktacode osk on määrittänyt niin sanotun todennäköisen asiakkaan. Se on pieni tai keskisuuri palvelualan yritys tai yksittäinen yrittäjä, joka tuottaa palveluita, joita voidaan varata.

Järjestelmän pitää pystyä edistämään sekä helpottamaan yrityksen liiketoimintaa. Yksi olennainen osa-alue edistämisessä on ajanvaraukseen käytettävän resurssin määrä, kun otetaan käyttöön ajanvarausjärjestelmä. Varausten vastaanottaminen tapahtuu osaksi automaattisesti. Tällöin on itse tuottavaan työhön enemmän resursseja käytettävissä. Yrityksissä, joissa työskentelee esimerkiksi enemmän hierojia tai kampaajia kuin itse yrittäjä, tekee ajanvarausjärjestelmä toiminnasta entistä organisoidumpaa. Jokainen merkkaa työaikansa itse, mutta kuka tahansa voi ottaa varauksen vastaan kenelle tahansa työntekijöistä.

5.2 Sidosryhmien tunnistus

Ensimmäisenä vaiheena ennen itse vaatimusmäärittelyn aloittamista on löytää sidosryhmät. Tätä varten pidimme aivoriihipalaverin Oktacode osk:n jäsenien kesken. Ideana oli miettiä ajanvarausjärjestelmän koko elinkaaren ajalta siihen mahdollisesti vaikuttavat sidosryhmät.



KUVIO 1. Sidosryhmät tuotteen elinkaaren aikana

Aivoriihessä paloiteltiin tuote elinkaaren ajalta määrittelyvaiheeseen, kehitysvaiheeseen ja valmiiseen tuotteeseen. Heti alussa huomattiin, että tuotteella on yksi sidosryhmä, joka on koko elinkaaren ajan mukana, ja se on project manager. Esitutkimus- ja määrittelyvaiheessa on mukana ohjelmistoarkkitehti, joka tekee suunnitelman järjestelmälle. Erityisesti esitutkimuksen aikana mukana on myös markkinatutkija. Esitutkimuksen aikana tehdään feasibility study, jonka tarkoituksena on selvittää, onko järjestelmää kannattava toteuttaa.

Tässä tapauksessa erityisesti, kun kyseessä on tuote, on tarpeellista tehdä jonkinasteista markkinatutkimusta ainakin alueellisesti. Keski-Suomen alueella on muutamia yrityksiä, joilla on tuotteena ajanvarausjärjestelmä. Kahteen näistä paneuduttiin tarkemmin. Siirryttäessä Ajavan kehitysvaiheeseen mukana on ohjelmistokehittäjiä ja testaajia. Vaatimusmäärittelyssä näillä sidosryhmillä on tärkeä rooli.

Valmiilla tuotteella on eniten eri sidosryhmiä. Ajavan ostaneella asiakkaalla on käyttäjänä sidosryhmä, joka voidaan vielä jakaa työntekijään ja ylläpitäjään. Luonnollisesti palvelun ostaneella asiakkaalla on omat asiakkaansa ja näistä muodostuu yksi sidosryhmä. Valmiille tuotteelle voidaan asettaa vielä muita sidosryhmiä. Näitä ovat myyjä ja itse tuotteen ylläpitäjä Oktacode osk:sta.

5.3 Vaatimusten kartoitus

Teoriaosuudessa tuli selväksi, että tapoja vaatimusten kartoitukseen on lukuisia. Ajavaa tehtäessä vaatimuksia saadaan tutkimalla jo olemassa olevaa versiota, mutta myös tutkimalla kilpailevia järjestelmiä. Aivoriihet Oktacoden sisällä ovat varmasti todella toimiva tapa löytää vaatimuksia ja erityisesti analysoida niitä. Sidosryhmille suoritettava kysely arvioi jo olemassa olevaa Ajavaa ja sen toimintoja.

Kysely

Tämä vaatimusmäärittelyn prosessi sujui hieman helpommin, kun vanhasta ajanvarausjärjestelmästä pystyttiin ottamaan mallia. Suurin osa vaatimuksista pystyttiin säilyttämään, mutta oli myös vaatimuksia, jotka pudotettiin pois. Ensimmäisenä askeleena kartoituksessa tehtiin kysely jo olemassa olevan ajanvarausjärjestelmän ominaisuuksista. Kyselyyn pyrittiin saamaan vastauksia kaikilta sidosryhmiltä. Kysely suoritettiin internetissä ja siihen lähetettiin kutsu joko sähköpostitse, puhelimitse tai henkilökohtaisesti tapaamalla.

Noin kymmenelle eri sidosryhmän jäsenelle lähetettiin kutsu ja kaikki vastasivat. Kyselyllä haluttiin selvittää, mitä ominaisuuksia vanhasta järjestelmästä säilytetään ja miten ominaisuudet priorisoidaan. Kysely oli hyvin yksinkertainen ja siihen oli helppo vastata. Siinä oli 32 kysymystä, joka on sama kuin jo olemassa olevan Ajava 1.0:n vaatimusten määrä. Sidosryhmän jäsenen piti arvioida vaatimuksen tärkeys välillä yhdestä viiteen, jossa viisi oli pakollinen ja yksi oli valinnainen.

Vastaukset olivat melko johdonmukaisia keskenään. Kuitenkin, kuten kuvitella voi, eri sidosryhmille eri vaatimukset saattavat olla prioriteetiltaan korkeampia. Liian eriäviä vastausmääriä yhteenkään kysymykseen ei tullut, joten niistä pystyttiin tekemään johtopäätökset vaatimusmäärittelyyn.

Vaatimuksen tärkeys

Arvioi vaatimuksen tärkeys asteikolla 1-5. (vaatimus on valmiissa järjestelmässä ominaisuus)

	1 (valinnainen)	2 (vähän tärkeä)	3 (melko tärkeä)	4 (erittäin tärkeä)	5 (pakollinen)
1. Asiakas voi rekisteröityä palveluun					
2. Asiakas voi muokata omia tietojaan					
3. Asiakas voi tunnistautua luotettavaksi käyttäjäksi (esimerkiksi tekstiviestipalveu, pankkitunnukset)					
4. Asiakas voi kirjautua järjestelmään					
5. Asiakas voi poistaa itsensä palvelusta halutessaan					
6. Asiakas voi tilata salasanan rekisteröinnissä annettuun sähköpostiosoitteeseen					
7. Asiakas voi selailla käynti- ja palveluhistoriaansa					
8. Ylläpitäjä voi lähettää järjestelmän kautta asiakkaille asiakaskirjeen					
9. Ylläpitäjä voi poistaa asiakkaan					
10. Ylläpitäjä voi muokata asiakkaan tietoja					
11. Ylläpitäjä voi lisätä asiakkaan					

Kuvio 2. Kysely sidosryhmille vaatimusten tärkeydestä

Haastattelut

Yksi parhaista metodeista löytää käyttäjävaatimuksia ovat haastattelut. Haastattelut oli järkevä kohdistaa vain osaan sidosryhmistä. Näitä olivat tulevan tuotteen käyttäjät: asiakas, työntekijä sekä ylläpitäjä. Todella moni käyttää tai on käyttänyt jotakin sähköistä ajanvarausjärjestelmää. Asiakkaiden löytäminen haastatteluun ei ollut vaikeaa. Haastatteluita oli yhteensä muutamia, joista yli puolet oli asiakkaita ja loput mahdollisia ylläpitäjiä tai työntekijöitä.

Vaatimusten runko oli jo valmiina olemassa olevan Ajavan takia, joten haastatteluissa keskityttiin lähinnä uusien innovaatioiden löytämiseen. Haastattelut olivat avoimia, jolloin niissä ei ollut edellä määriteltyä agendaa tai kysymyssarjaa.

Aivoriihet

Ajava tulee olemaan Oktacoden tuote, joten Oktacoden jäsenet päättävät loppujen lopuksi sen ominaisuuksista. Tuote suunnitellaan, kehitetään, myydään ja ylläpidetään yrityksen toimesta, joten suurin osa sidosryhmistä löytyy sieltä. Juuri siksi keskusteluita käytiin yrityksen sisällä. Tuotteen myyjillä on täysin erilaisia tarpeita tuotteen suhteen kuin esimerkiksi ohjelmistoarkkitehdilla.

Näissä tapaamisissa keskusteltiin vapaasti mutta kuitenkin siten, että kaikilla oli tavoitteet Ajavan suhteen tiedossa. Kaikki ajatukset laitettiin ylös, jotta niistä voitiin jälkikäteen poimia olennaiset ideat.

Vastaavanlaisten järjestelmien tutkiminen

Yksi antoisa tapa löytää vaatimuksia Oktacoden tuotteelle oli tutkia kilpailijoiden varausjärjestelmiä. Tutkittavia asioita olivat järjestelmän rakenne, vahvuudet ja heikkoudet. Tutkittavia järjestelmiä löydettiin seitsemän kappaletta. Yleiseksi huomioksi lähes heti ilmeni järjestelmien rakenne. Suurin osa järjestelmistä oli tehty ”software as a service” -muottiin. Kovinkaan monen ajanvaraus ei siis löytynyt itse palveluita tuottavan yrityksen omalta palvelimelta.

Johtuen edellisestä, ajanvarauksen ulkoasullinen yhteneväisyys itse sivujen kanssa ei ollut kaikissa järjestelmissä kovinkaan luonnollista. Seitsemästä tutkitusta kilpailevasta varausjärjestelmästä puolella oli jotain viittauksia siitä, että oli yritetty saada ajanvaraus yhteneväiseksi ulkoasullisesti. Parhaita olivat ne, joissa ajanvaraus on sivujen sisällä. Näistä ei kuitenkaan ollut varmaa tietoa, olivatko ne tehty software as a service -mallin mukaisesti.
Noin puolet järjestelmistä vaati rekisteröitymistä. Suurimmassa osassa se oli hoidettu perinteisesti lomakerekisteröitymisellä. Yhdessä järjestelmässä rekisteröityminen oli liitetty käyttäjän matkapuhelimeen. Käyttäjätunnuksena toimi matkapuhelinnumero. Tämä kasvattaa asiakkaiden luotettavuutta runsaasti. Rekisteröitymisen pakko on osittain hyvä asia. Ajavaan oli ajateltu rakentaa rekisteröityminen siten, että se ei ole pakollista mutta nopeuttaa

käyttäjän varauksen tekoa. Ensikertalainen voi joko tehdä varauksen ilman rekisteröintiä tai rekisteröityä samalla.

Tutkittavissa järjestelmissä tämä rekisteröitymisvaihtoehto tuntui järkevimmältä. Seitsemän järjestelmän joukossa oli vain kaksi, joissa ajanvaraus oli upotettu internetsivuihin toimivalla tavalla. Molemmissa toiminnallisuudetkin olivat hyvällä tasolla. Oktacode hakee ehdottomasti vastaavanlaista ratkaisua. Ajavan on oltava yksilöity, mutta toimiva. Voidaan jopa sanoa, että tämän hetken käytössä oleva Ajava on käytettävyydeltään parempi kuin suurin osa kokeilluista järjestelmistä. Kun käytettävyys ja toiminnallisuudet pidetään samalla tai paremmalla tasolla ja Ajavan runkojärjestelmä toimii SaaS-mallin mukaisesti, niin se tulee olemaan kilpailukykyinen tuote.

5.4 Vaatimusten analysointi

Vaatimuksia kerättiin erilaisilla metodeilla. Sidosryhmiä haastateltiin, sidosryhmille tehtiin kyselyitä, yrityksen sisällä pidettiin aivoriihiä ja kilpailevia järjestelmiä analysoitiin. Kaikilla näillä tavoilla löydettiin erilaisia vaatimuksia ajanvarausjärjestelmälle. Vaatimukset pitää kuitenkin analysoida. Tähän prosessiin liittyi joukko erilaisia toimintoja.

Vaatimusten tarpeellisuus, toimitettavuus ja toistuvuus

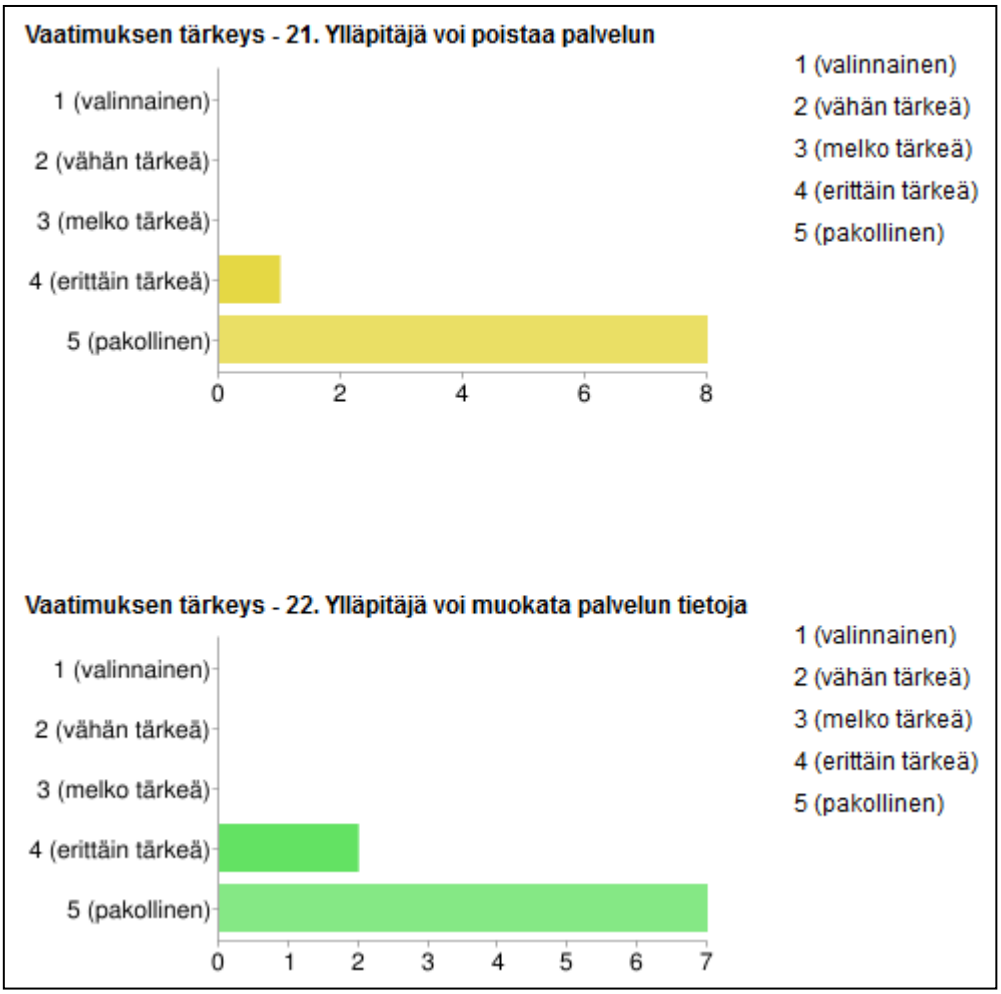
Lista vaatimuksista oli pitkä, joten oli oletettavaa, että sieltä löytyy niin sanotusti turhia vaatimuksia. Heti ensimmäiseksi tuli selvittää, edistääkö jokainen vaatimus oletettavan asiakkaan liiketoimintaa tai tuoko se vastauksen johonkin ongelmaan. Tällä perusteella voitiin miettiä samalla itse vaatimuksen tarpeellisuutta tai ainakin sen prioriteettia. Suurin osa vaatimuksista periytyi edellisestä Ajavasta. Tämän vuoksi kovin montaa turhaa vaatimusta ei löytynyt.

Toinen tärkeä asia, joka analyysivaiheessa tehtiin, oli vaatimuksen toimitettavuuden tutkiminen. Joitakin vaatimuksia jouduttiin pudottamaan pois, koska niiden toteuttamisen työmäärä olisi ollut liian suuri hyötyyn nähden. Järjestelmän rakentamiseen varataan tietty määrä resursseja, ja ne on käytettävä ytimekkäästi. Tämän takia vaatimukset pitää pystyä mitoittamaan oikein resurssien määrälle ja mahdolliselle budjetille. Ajavan vaatimusten analysoinnin alkupään vaiheet olivat kohtuullisen helppo suorittaa. Suurin syy tähän oli vahva vaatimusten pohja jo toimivasta järjestelmästä.

Vaatimuksista piti vielä selvittää niiden mahdollinen toistuvuus ja se, löytyykö epätäydellisiä vaatimuksia. Samoja vaatimuksia paljastui jonkin verran, kun tutkimiseen käytettiin aikaa. Näissä tapauksissa duplikaatit poistettiin ja jäljelle jätettiin luonnollisesti yksi samanlainen vaatimus. Vaatimukset olivat yleisellä tasolla kirjoitettu niin hyvin, että epätäydellisiä vaatimuksia, joista puuttuisi tarvittavia rajoitteita, ei löydetty.

Vaatimusten priorisointi

Aiemmin esille tullut kuvio näytti, miten eri sidosryhmät antoivat oman mielipiteensä vaatimusten tärkeydestä. Tämä oli pohja järjestelmän vaatimusten priorisoinnille. Ennen vastauksia oletettiin vastausten sisältävän todella vaihtelevia kantoja vaatimusten prioriteeteista. Yllätykseksi- eri sidosryhmien vastauksista pystyttiin tekemään johtopäätös jokaisen vaatimuksen kohdalla.



KUVIO 3. Sidosryhmille esitetyn prioriteettikyselyn vastauksia

Esitetyssä kuviossa nähdään kahden eri vaatimuksen vastaukset. Vaatimuksia palvelun poistamisesta ja palvelun tietojen muokkaamisesta pidetään tärkeinä kaikkien sidosryhmien kesken. Vaatimusmäärittelyn dokumentissa prioriteettitasojen määrä laskettiin viidestä kolmeen. Vaatimusmäärittelyssä prioriteettitasot ovat tärkeä, keskinkertainen ja matala. Vaatimus on tärkeä, kun se on olennainen osa käyttökelpoista järjestelmää. Keskinkertainen se on silloin, kun se parantaa järjestelmän käyttökelpoisuutta huomattavasti. Järjestelmä kuitenkin toimii ilman sitä, mutta suurin osa näistä vaatimuksista toteutetaan. Matala vaatimus tuo jo käyttökelpoiseen järjestelmään lisäarvoa. Nämä vaatimukset voidaan mahdollisesti toteuttaa myöhemmissä versioissa.

Vaatimukset jaoteltiin eri ryhmiin. Priorisoinnin lisäksi toiminnalliset vaatimukset jaettiin kolmeen ryhmään käyttäjäryhmien mukaisesti. Kolme ryhmää ovat asiakas, työntekijä ja ylläpitäjä.

Ei-toiminnalliset vaatimukset ja laadulliset tavoitteet

Järjestelmävaatimukset ja ei-toiminnalliset vaatimukset tulivat kaikki Oktacodelta. Nämä vaatimukset sisältävät paljon huomioitavia asioita. Seuraavana käsitellään Oktacoden määrittelemät vaatimukset. Yksi olennaisimpia asioita on ohjelmointikieli. Oktacoden jäsenet päättivät yhdessä, että ohjelmointikieli on backend-puolella PHP ja frontend-puolella PHP sekä Javascript ja sen erinäiset kirjastot. Ajanvarausjärjestelmän käyttöön vaaditaan kahta palvelinta. Toinen on Oktacoden ylläpitämä backen-dpalvelin, jonka on sisällettävä MySQL, PHP- ja Javatuki. Toinen palvelin on asiakkaan ja sieltä on löydyttävä tuki PHP–ohjelmointikielelle.

Ohjelmointikielen ja tietokannan versiot on myös todettava, koska ne kehittyvät jatkuvasti. PHP:n uusin versio on vaatimusmäärittelyn hetkellä 5.3.8 ja MySQL:n versio on 5.5.18. Versioista tulee kaksi palvelinvaatimusta, joissa todetaan, että järjestelmän on tuettava vähintään viimeisimpiä versioita. Nämä vaatimukset oli osoitettu asiakkaan ja Oktacoden omille palvelimille. Vaatimuksia tehtiin myös järjestelmän frontendin toiminnalle. Järjestelmän tuki selaimille on tärkeä löytyä vaatimuksista. Oktacoden sisällä päätettiin, että järjestelmän tulee tukea kuutta yleisintä työpöytäselainta. Yleisimmät työpöytäselaimet ja niiden versiot ovat:

- Google Chrome versio 8
- Mozilla Firefox 8
- Internet Explorer 8
- Safari 5
- Opera 10
- SeaMonkey 5.

Selainversiot eivät välttämättä ole viimeisimpiä. Sopivat versiot on analysoitu käyttäjämäärien ja julkaisupäivämäärän perusteella. Esimerkiksi w3schoolsin

(2011) mukaan marraskuussa 2011 Internet Explorerin käyttömäärä on 21.2 % kaikista selaimista. Näistä 21.2 prosenttiyksiköstä 11.5 prosenttiyksikköä on IE8–käyttäjiä. Sitä vanhemman IE7:n käyttäjämäärä on enää 3.4 prosenttiyksikköä 21.2 prosenttiyksiköstä. Internet Explorer 7 on julkaistu vuonna 2006, joten Oktacode osk ei enää näe järkeväksi toteuttaa sille täydellistä tukea.

Ajavan mahdolliset mobiilikäyttäjät haluttiin myös huomioida vaatimusmäärittelyssä. Suomessa on vuonna 2011 käytetty neljää suurempaa mobiiliselainta. Näistä neljä on prosentuaalisilla käyttäjämäärillään muita edellä. Androidin, iPhonen ja Nokian selainten käyttäjämäärät ovat yhteensä noin 80 %. Neljänneksi käytetyin mobiiliselain on Opera, jonka käyttäjämäärä on 12 %. (StatCounter Global stats, 2011).

Näitä statistiikkoja peilaten järjestelmän on toimittava vähintään seuraavilla mobiiliselainten versioilla:

- Android 4.0
- Safari 5
- Nokia Browser 7.1
- Opera Mobile 11.5.

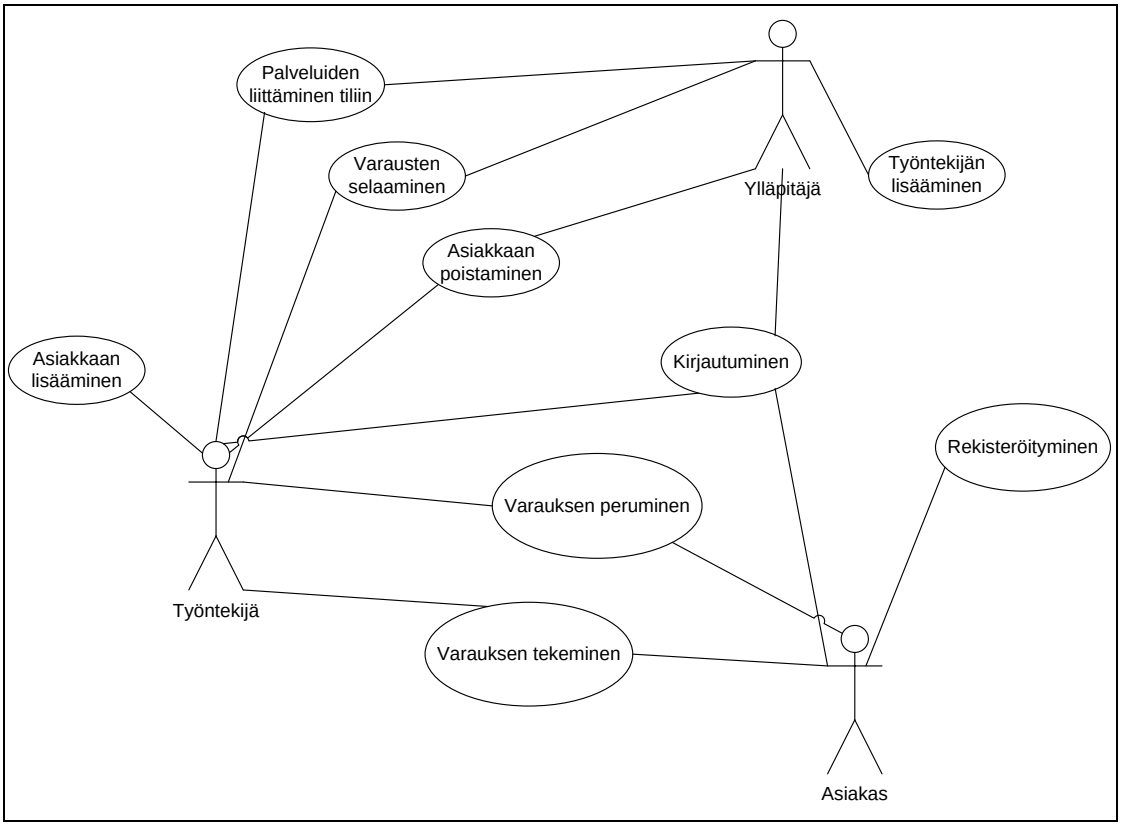
Oktacode osk asetti järjestelmälle myös laadullisia tavoitteita. Nämä poikkeavat hieman vaatimuksista epätarkkuutensa vuoksi. Yhtenä tavoitteena on rakentaa käyttöliittymät siten, että ne ovat käyttäjäystävällisiä ja loogisia. Yksi pohdittava asia oli kieli. Alustavasti päädyttiin ratkaisuun, jossa Ajavan kielenä on suomi, mutta järjestelmä on rakennettava siten, että se voidaan helposti laajentaa mille tahansa kielelle. Koodin toteuttamiseenkin otettiin kantaa siltä osin, että se pitää dokumentoida ja kommentoida kattavasti jatkokehityksen helpottamiseksi.

Laadullinen tavoite on myös testauksen suorittaminen yksikkö-, integrointi- ja järjestelmävaiheissa. Järjestelmän laatua varmistetaan lisäksi katselmoinneilla. Niitä suoritetaan vaatimusmäärittelydokumenttiin.

Turvallisuusajattelun kannalta todettiin, että kaikissa järjestelmän syötekentissä on noudatettava sopivaa syötetiedon tarkistusta. Asiakkaan kannalta ajateltiin siirrettävyyttä ja poistamista. Järjestelmän tulee olla helposti siirrettävissä toiselle palvelimelle. Järjestelmän tulee olla myös helposti poistettavissa frontend–palvelimelta.

Käyttötapaukset

Oktacodelle tehdyssä vaatimusdokumentissa on otettu huomioon eri sidosryhmiä tekemällä käyttötapauskaavio ja sen lisäksi käyttötapaukset tärkeimmistä vaatimuksista.



KUVIO 4. Ajavan käyttötapauskaavio

Käyttötapauskaavio antaa yleiskuvan toteutettavasta järjestelmästä. Tämä voi olla esimerkiksi ohjelmoijille tärkeä osa, joka selventää järjestelmän toimintaa. Kaaviosta löytyvät kaikista tärkeimmät toiminnot, joita eri käyttäjäryhmät suorittavat. Ajanvarausjärjestelmässä tulee olemaan iso joukko muitakin toimintoja.

Itse käyttötapaukset kirjoitettiin järjestelmän kannalta tärkeimmistä toiminnoista. Yhtenä olennaisimmista toiminnoista pidettiin ajan varaamista. Esimerkiksi tästä tehtiin käyttötapaus.

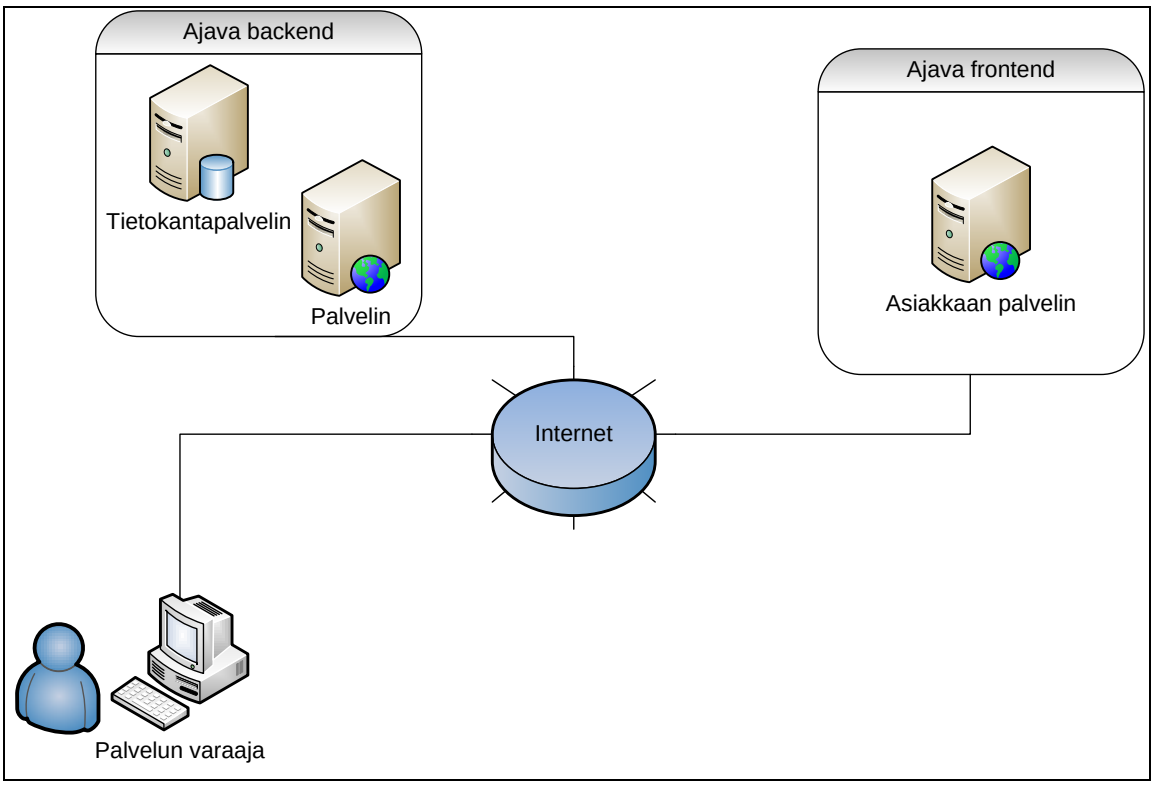
TAULUKKO 1. Käyttötapaus 4: ajan varaaminen

Käyttötapaus 4: Ajan varaaminen / ylläpitäjä, työntekijä
Suorittaja: Ylläpitäjä, Työntekijä
Käytettävyysvaatimukset: Käyttäjä voi varata asiakkaalle palvelun ja mahdollisen tuottajan asiakkaan tarpeiden mukaan.
Alkuehdot: Käyttäjä on kirjautunut järjestelmään.
Kuvaus: Käyttäjä valitsee asiakkaan, palvelun, mahdollisen tuottajan ja ajankohdan. [Poikkeus: Uusi asiakas]. Käyttäjä tekee varauksen järjestelmään.
Poikkeukset: Uusi asiakas: Käyttäjä siirtyy uuden asiakkaan lisäämiseen.
Loppuehdot: Varattu aika on tallennettu järjestelmään.
Täyttää vaatimukset: Y12, Y1

Järjestelmäarkkitehtuuri

Vaatimusmäärittelyssä otettiin luonnollisesti kantaa myös järjestelmäarkkitehtuuriin. Kuten jo aiemmin on tullut esille, järjestelmän backend ja frontend rakennetaan PHP-web-ohjelmointikielellä. Järjestelmä noudattaa MVC-mallia, jossa järjestelmä on jaettu kolmeen kerrokseen: view, controller ja model. View eli näkymäkerroksessa haluttiin käytettäväksi XHTML-, HTML- ja Javascript–kieliä. Controller- ja model–kerrokset rakennetaan suurimmaksi osaksi PHP:lla. Model eli mallikerroksessa voidaan käyttää jonkin verran SQL–tietokantakieltä.

PHP:lla on paljon erilaisia frameworkeja, jotka edistävät kielen oliopiirteitä ja tuovat lisäkirjastoja ja laajuuttaa koodiin. Oktacode on käyttänyt jo useassa projektissa Codeigniter PHP–frameworkia, joten se oli luonnollinen valinta tähänkin tuotteeseen. Kuten aiemminkin on tullut ilmi, järjestelmä jaetaan backendiin ja frontendiin.



KUVIO 5. Ajavan rakenteellinen arkkitehtuurikaavio

6 TUTKIMUKSEN TULOKSET

Tutkimus eteni oikeastaan hyvin samalla tavalla, kuin etukäteen oli pystynyt kuvittelemaan. Tutkimuksen teorian pääpaino oli vaatimustenhallinnan osalta juuri sen alkupäässä eli vaatimusmäärittelyssä. Se sopi oikeastaan tähän tutkimukseen hyvin. Työn helpottamiseksi oli ennen tutkimusta määritelty tutkimuskysymyksiä, joiden tarkoitus oli tuoda vastauksia tutkimusongelmaan. Tutkimusongelmana oli se, miten voidaan tehdä vaatimusmäärittely ajanvarausjärjestelmälle.

Ajanvarausjärjestelmän vaatimukset

Ensimmäinen tähän ongelmaan ratkaisua hakeva kysymys koski ajanvarausjärjestelmän vaatimuksia. Mitkä ovat tulevan ajanvarausjärjestelmän vaatimukset? Vastaus saatiin hyvin kokonaisvaltaisesti, ja tutkimus oli onnistunut tämän tutkimuskysymyksen

kannalta. Käytännössä vastaus kysymykseen nähdään Ajavan vaatimusmäärittelyn dokumentissa, joka on liitteenä tutkimuksessa. Vaatimuksia kertyi yli neljäkymmentä, jos mukaan lasketaan toiminnallisten käyttäjävaatimusten lisäksi ei-toiminnalliset sekä laadulliset vaatimukset. Vaatimuksia kerättiin melko kirjavalta joukolta eri sidosryhmien edustajia, joten voidaan tehdä myös johtopäätös, että tutkimuskysymykseen vastasi joukko.

Vaatimukset on suurimmaksi osaksi kerätty tutkitusta teoriasta löytyvien menetelmien avulla. Tämän perusteella voidaan ajatella, että teoriaosio todella tukee tutkimusta. Pitää muistaa, että tutkimuskysymys ei liity ainoastaan vaatimusten keräämiseen vaan vaatimuksiin kokonaisuudessaan. Kysymys haluaa vastaukseksi listan vaatimuksista, jotka on analysoitu, priorisoitu, kategorisoitu ja validoitu. Nämä kaikki prosessin askeleet on otettu huomioon vaatimuksissa.

Vaatimusmäärittelyn prosessi ajanvarausjärjestelmässä

Yhtenä tutkimuskysymyksen oli vaatimusmäärittelyn prosessi ja se, minkälainen on juuri sopiva tähän ajanvarausjärjestelmään. Tutkimuksen toteutusosiossa ei suoranaisesti vastata tähän kysymykseen. Tutkimuskysymys kuitenkin oli vahvasti mukana koko tutkimuksen ajan. Toteutusosio etenee synkronisesti, ja samalla siinä kuvataan vaatimusmäärittelyn prosessia. Voidaan siis todeta, että tämä tutkimuskysymys sai kattavan vastauksen ja ongelmaan löytyi ratkaisu. Pääasiallisesti tämän vaatimusmäärittelyn prosessi eteni samalla tavalla- kuin teoriassa.

Jotkut asiat jouduttiin tekemään poikkeamalla tutkitusta teoriasta. Tämä johtui siitä, että kyseessä ei ollut aivan perinteinen asiakaslähtöinen järjestelmäprojekti, vaan yrityksen tuote, jota tulee käyttämään kolmas osapuoli. Esimerkiksi vaatimusten kartoitusvaiheessa vaatimusten keräys ei tapahtunut kokonaan teoriasta löytyneillä keinoilla, vaan siinä sovellettiin myös uusia keinoja. Kartoituksessa käytettiin perinteisiä keinoja, kuten aivoriihet, haastattelut sekä kilpailevien tuotteiden tutkiminen. Toisaalta käytettiin myös kyselyä, joka oli tehty jo olemassa olevan Ajavan vaatimuslistan pohjalta.

Tämä useiden eri sidosryhmien kesken tehty kysely auttoi vaatimusten kartoitusvaiheessa sekä vaatimusten analysointivaiheessa. Sen avulla oli helppo jättää vaatimuksia pois ja priorisoida ne.

Teoriassa esille tullut fakta, jossa todetaan vaatimusmäärittelyn olevan syklinen tapahtuma, pitää paikkansa. Siinä saatetaan palata uudestaan ja uudestaan prosessin osasta toiseen, kunnes määrittely on valmis. Ainoastaan liiketoiminnan tavoitteiden ja ongelmien pohtiminen tapahtui vain kerran ennen vaatimusmäärittelyn aloittamista. Suoritettujen prosessin osien taustalla rakentui samaan aikaan vaatimusmäärittelyn dokumentti. Se tulee olemaan mittari onnistuneelle vaatimusmäärittelylle, kun itse järjestelmää ruvetaan rakentamaan.

Koska kyseessä on tuote, jouduttiin myös eri prosessien osille antamaan painoarvoa. Esimerkiksi kilpailevien tuotteiden tutkiminen ja analysointi oli erityisen tärkeä vaihe rakennettavan tuotteen menestyksen kannalta. Se toi joitakin yksittäisiä vaatimuksia lisää, mutta suurin hyöty oli isompiin linjauksiin järjestelmän rakenteen kannalta. Toisaalta esimerkiksi vaatimusten lyöminen täysin lukkoon vaatimusmäärittelyn jälkeen ei ollut välttämättömyys. Oktacode osk tulee omistamaan tuotteen ja on vastuussa kehityksestä itse, joten mahdollista asiakkaan tyytyväisyyttä ei tarvitse jatkuvasti suunnittelu- ja kehitysvaiheessa ottaa huomioon. Toisaalta tämä tuo myös lisänpainoarvoa vaatimusten hallinnoinnille, mikäli vaatimuksia vielä muutetaan.

Ajanvarausjärjestelmän sidosryhmät

Kolmannen tutkimuskysymyksen tarkoitus oli löytää ajanvarausjärjestelmän sidosryhmät. Sidosryhmien löytäminen ja tunnistaminen on liiketoiminnan tavoitteiden ja ongelmien ohella tärkein asia ennen vaatimusmäärittelyn aloittamista. Sidosryhmien löytäminen ei ollut nopea prosessi. Aivoriihet Oktacoden jäsenten kesken oli hyvä keino etsiä sidosryhmiä. Käytännössä erilaisten miellekarttojen rakentaminen aivoriihissä tuotti tulosta. Jotta sidosryhmät pystyttiin löytämään, piti miettiä kaikkia osapuolia tulevan tuotteen elinkaaren ajalta.

Teoriaosuudessa kävi selväksi, että tämä on tärkeä osa vaatimusmäärittelyä. Tämän asian kanssa toimittiinkin juuri niin. Sidosryhmiä löytyi yhteensä yhdeksän, joista suurinta osaa pystyttiin kuulemaan vaatimusmäärittelyssä. Vaatimusmäärittely olisi varmasti onnistunut kapealla sidosryhmien mukaan ottamisellakin, mutta lopputulos olisi jäänyt myös kapeaksi. Jos toiminnalliset käyttäjävaatimukset olisi keksitty pelkästään Oktacoden sisällä, lopputuote tuskin vastaisi niin hyvin tarkoitetun käyttäjäryhmän tarpeita. Sidosryhmät olivat erittäin tärkeä osa vaatimusten kartoittamisen ja priorisoinnin kannalta.

7 JOHTOPÄÄTÖKSET JA POHDINTA

Tutkimukseen voi olla tyytyväinen, kun sen tavoite tai määränpää toteutuu. Tutkimuksen tarkoitus oli luoda vaatimusmäärittely ajanvarausjärjestelmälle, jonka pohjalta toimeksiantaja voi rakentaa kilpailukykyisen tuotteen. Alustavasti vaikuttaa siltä, että toimeksiantaja on saanut, mitä on pyytänyt. Tutkimus eteni suunnitelmien mukaisesti. Ainut merkittävästi suunnitelmasta poikennut tekijä oli aikataulu. Toimeksiantaja ei antanut mitään aikakehyksiä työn valmistumiselle, mutta itse määriteltyyn aikatavoitteeseen ei päästy.

Teoria tuki käytännön osuutta eli vaatimusmäärittelyä hyvin. Joitakin teorian osuuksia ei voitu hyväksikäyttää niin paljon kuin muita. Tämä johtuu suurimmaksi osin vaatimusmäärittelyn erilaisuudesta eri tilanteissa. Toimeksiantaja haluaa rakentaa tuotteen, mutta suurimmassa osassa teorialähteistä vaatimustenhallinta kertoo, miten asiat tehdään isoissa järjestelmäprojekteissa. Teoriasta on kuitenkin pystytty poimimaan olennaiset ja sopivat metodit tähän työhön. Niiden soveltaminen hieman eri kehyksiin toi tutkimukseen lisähaastetta. Työssä tuli erilaista näkökulmaa erityisesti vaatimusmäärittelyyn.

Teoria ja itse tutkimuksen toteutus antavat arvokasta oppia tietynlaisen järjestelmän suunnitteluun juuri tietynlaisessa ympäristössä. Tutkimuksesta voi olla apua erityisesti tilanteisiin, jossa joudutaan toimimaan

vaatimustenhallinnassa normaalista poikkeavalla tavalla. Teoriaosuudessa kävi ilmi, että tiettyä vaatimustenhallinnan prosessia ei voi yleistää kaikkiin organisaatioihin. Tämän työn tutkimusosuus tukee kyseistä väitettä. Poiketen joistain perinteisistä asiakaslähtöisistä järjestelmäprojekteista otettiin sidosryhmät tarkasti huomioon vaatimusmäärittelyssä.

Opinnäytetyössäni käytettiin tutkimusmenetelminä case- eli tapaustutkimusta. Tapaustutkimus soveltui mielestäni hyvin opinnäytetyöhön. Opinnäytetyö noudattelee case-tutkimuksen prosessin periaatteita. Käsitteellis-teoreettinen tutkimus on toiminut myös omalla tavallaan opinnäytetyössä, koska tulokset tuottivat uniikin vaatimusmäärittelyn ja uuden prosessimallin vaatimusmäärittelyyn.

Opinnäytetyössä tulokset on johdettu vaatimusmäärittelyn kohdalla joko käyttäjävaatimuksista tai toimeksiantajan omista vaatimuksista. Vaatimukset saatiin eri sidosryhmiltä haastattelemalla sekä suorittamalla kyselyitä. Tulosten luotettavuutta on syytä tarkastella kriittisesti. Kyselyt suoritettiin kontrolloidusti ja kaikki vastaajat valittiin etukäteen. Sidosryhmiä oli yhteensä yhdeksän, ja näistä suurinta osaa haluttiin kuulla kyselyn muodossa. Kyselyiden tuloksiin ei ollut pääsyä muilla kuin opinnäytetyön tekijällä ja toimeksiantajalla. Tutkimuksen tuloksia on käsitelty siis luotettavasti ja tutkimuksen eettisiä periaatteita noudattaen. Sidosryhmiltä saadut tulokset myös analysoitiin työn tekijän ja toimeksiantajan puolesta. Vastauksia voidaan pitää laadukkaina, koska vastaajat oli etukäteen valittu sekä ohjeistettu kyselyyn.

Opinnäytetyön aiheena oli vaatimustenhallinta, joka on hyvin ajankohtainen ja tärkeä aihe ohjelmistotuotannossa. Vaatimustenhallinta on tärkeä toimeksiantajalle, mutta erityisesti opinnäytetyön tekijälle henkilökohtaisesti. Siinä vaiheessa, kun näin ensimmäiset teokset, jotka käsittelivät requirements engineering aihetta, tajusin sen olevan todella tärkeä alue ohjelmistotuotantoa. Vaatimustenhallinnasta puhutaan aina paljon ja se on kaikkialla ajankohtaista, mutta tuntuu silti, että siinä tehdään paljon virheitä. Uskon, että vaatimustenhallinnalla on suuri merkitys työnhaussa.

Opinnäytetyö käsittelee kuitenkin vain osaa vaatimustenhallinnan alueesta. Vaatimusmäärittely on totta kai tärkeä osa vaatimustenhallintaa, mutta se on osa, joka suoritetaan ennen ohjelman suunnittelu- ja rakennusvaihetta. Yksi osa-alue on jäänyt käsittelemättä lähes kokonaan. Alue on vaatimusten hallinnointi eli requirements management. Tämä osa-alue on mukana, mikäli vaatimukset muuttuvat kesken vaatimusmäärittelyn tai jopa sen jälkeen. Hallinnointi määrittelee prosessin, sen, mitä tapahtuu, kun vaatimus poistetaan, muutetaan tai lisätään.

Tutkimusta aiheesta olisi helppo jatkaa, kun projekti etenee suunnittelu- ja rakennusvaiheeseen ja on lopulta valmis. Teoriassa kävi jo aiemmin ilmi, että vaatimustenhallinta on liitettynä järjestelmään sen koko elinkaaren ajan. Vaatimusmäärittelyn onnistumisen arviointi onnistuu myös helpommin, kun nähdään sen rooli kokonaisuudessa. Tästä löytyy siis hyvä aihe jatkotutkimuksen kannalta.

LÄHTEET

Aouad, G. & Arayici, Y. 2010.Requirements Engineering for Computer Integrated Enviroments in Construction. Blackwell Publishing.

Berenbach, B., Kazmeier, J., Paulish, D. & Rudorfer, A. 2009. Software & Systems Requirements Engineering In Practice. The McGraw-Hill companies.

Browser statistics. N.d. W3schoolsin tuottama taulukko käytetyimmistä selaimista vuonna 2011. Viitattu 30.12.2012. http://www.w3schools.com/browsers/browsers_stats.asp

Case-tutkimus. N.d. Jyväskylän ammattikorkeakoulu. Viitattu 10.1.2012. http://www.amk.fi/opintojaksot/0709019/1193463890749/1193464144782/1194348546586/1194356433452.html

Järvinen, A. & Järvinen, P. 2000. Tutkimustyön metodeista.Tampere:Opinpajan kirja.

Kotonya, G. & Sommerville I. 1998. Requirements Engineering.

Le Vier, D. 2010. Writing Software Requirements Specifications. http://techwhirl.com/articles/writingsoftwarerequirementsspecs/

Mobile browser statistics. 2011. StatCounter Global Statsin tuottama lista Suomen käytetyimmistä mobiiliselaimista 2010-2011. Viitattu 30.12.2011. http://gs.statcounter.com/

Young, R. 2004. The Requirements engineering handbook. Artech House. Norwood.

LIITTEET

Liite 1. Vaatimusmäärittelyn dokumentti

Johdanto

Yleiskuvaus

Oktacode osk:lla on tällä hetkellä tehty ajanvarausjärjestelmä, joka on käytössä yhdellä asiakkaalla. Järjestelmä, joka on nimeltään Ajava, on räätälöity yhden tietyn asiakkaan tarpeisiin, joten se ei ole tällä hetkellä helposti monistettavissa.Tällä hetkellä ajanvarausjärjestelmä vaatii myös aina oman tietokannan ja palvelimen toimiakseen. Tarkoituksena on kuitenkin tehdä myytäväksi tarkoitettu tuote, jota on helppo monistaa eri aloille tarpeesta riippuen. Tämän hetkinen Ajava on tehty tukemaan vain palveluita, joissa ihminen on resurssina. Esimerkkinä hieroja tai terapeutti. Tästä poiketen halutaan myös seuraavan Ajavan tukevan esimerkiksi liikuntahalleja, hotelleja, autovuokrausta tai mitä tahansa, jossa varataan jotakin resurssia tai palvelua joksikin ajaksi.

Ajava 2.0:n halutaan myös tukevan mobiilikäyttöä. Ensimmäisessä kehitysversiossa ei erinäistä applikaatiota ruveta toteuttamaan varausta varten, vaan web-pohjainen frontend on luotava siten, että se toimii myös yleisimmillä mobiiliselaimilla.

Lukijtaulukko

Seuraava taulukko kertoo kenen kannattaa ja tulisi lukea vaatimusdokumentti.

Lukijaryhmä	Miksi lukea?
Projektipäällikkö	Ymmärtää vaatimukset, jotta voi hallita projektia.
Arkkitehti	Ymmärtää vaatimukset, jotta voi suunnittella järjestelmän arkkitehtuurin.

Asiakas	Voi hyväksyä ja antaa palautetta vaatimuksista.
Järjestelmäkehittäjä	Ymmärtää mitä toiminnallisuuksia ja ominaisuuksia järjestelmän pitää ymmärtää.
Testaaja	Osaa testata järjestelmää peilaten vaatimuksia.

Sanasto

Sana Selitys

SaaS Software as a Service (SaaS) tarkoittaa ohjelmiston hankkimista palveluna perinteisen lisenssipohjaisen sijasta. Asiakaskohtaisia tuotantoympäristöjä ei ole, vaan sama tuotantoympäristö palvelee useampaa tai kaikkia asiakkaita. Asiakkaat käyttävät SaaS-ohjelmistoa yleensä Internet-selaimella.

Frontend Asiakaspuolen osa järjestelmästä, joka toimii yhteistyössä järjestelmän backendin kanssa.

Backend Suurin osa järjestelmän toiminnoista tapahtuu täällä. Tietokanta toimii yhteistyössä backend-ohjelman kanssa. Frontend ja backend ovat yhteydessä toisiinsa.

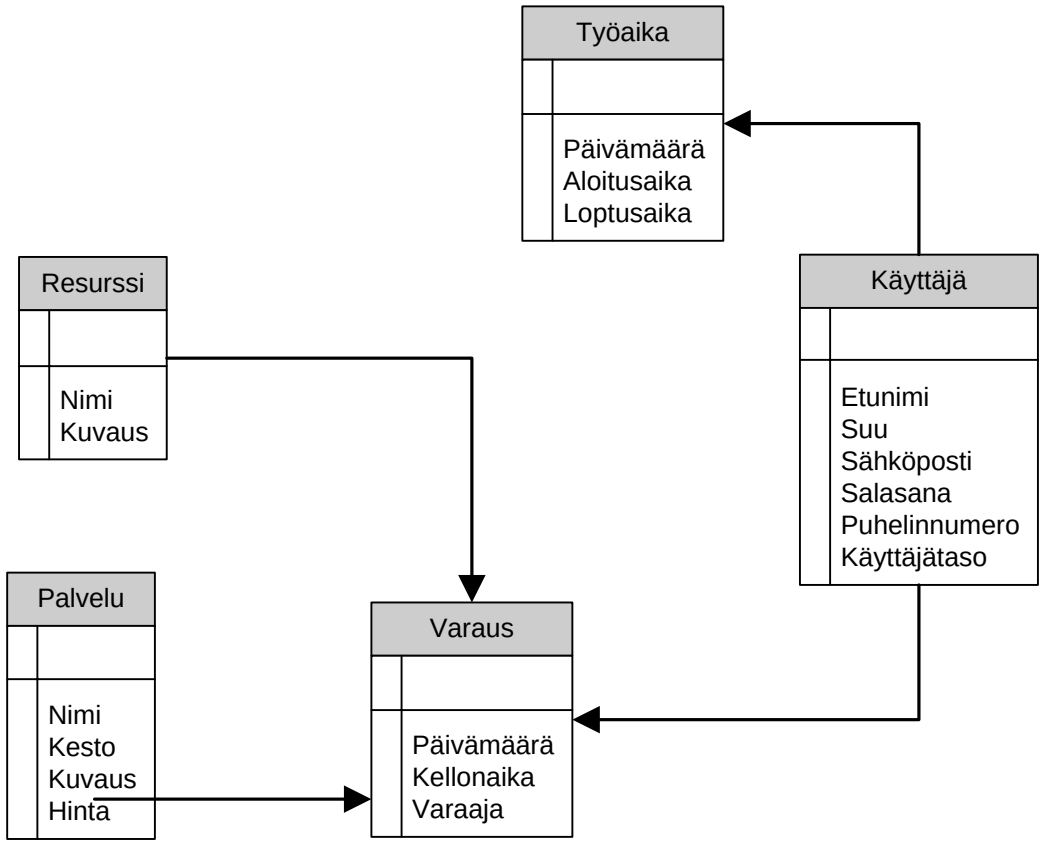
JavaScript Web-selaimen oliopohjaimen komentosarjakieli

MVC Model-View-Controller on ohjelmistotuotannossa käytettävä arkkitehtuurimalli, jossa ohjelmisto jaetaan kolmeen osaan: tietosisällön hallinta, käyttöliittymä sekä näiden välissä toimiva kontrolliosio.

MySQL Avoimen lähdekoodin olio-relaatiotietokantajärjestelmä. MySQL-kantaa voidaanhallinnoida SQL-kielellä

Käyttötapaus Käyttötapauksessa kuvataan järjestelmän ja sen käyttäjän välinen vuorovaikutus, jossa käyttäjä pyrkii saavuttamaan jonkin tavoitteen käyttämällä järjestelmää.

Järjestelmän tietosisältö



Kuva 1 järjestelmän tietosisältö

Vaatimukset

Järjestelmän vaatimukset on jaettu pääluokkiin. Näitä ovat ei-toiminnalliset, toiminnalliset ja laadulliset vaatimukset. Lisäksi dokumentissa on käyty läpi hieman teknisiä vaatimuksia itse käyttäjältä. Toiminnallisissa vaatimuksissa on kerrottu myös prioriteettitaso.

Järjestelmävaatimukset

Järjestelmä on jaettu frontendiin ja backendiin. Frontend on asiakkaalle näkyvä osa, jolla on yhteys backendiin. SaaS-systeemillä on toinenkin ratkaisuvaihtoehto, jossa kaikki asiakkaat ottavat ilman frontendiä yhteyden suoraan palveluntuottajan backend-järjestelmään. Tällaisissa tapauksissa ajanvarausjärjestelmän muokkaaminen asiakkaan näköiseksi on yleensä vaikeampaa ja saattaa vaikuttaa kömpelöltä.

Asiakkaalle räätälöitävä frontend on ratkaisu, jota tässä järjestelmässä haetaan. Se on helppo asentaa ja asiakkaalta ei vaadita muuta kuin palvelin, jossa esimerkiksi jo nykyinen sivusto on.

Ei-toiminnalliset vaatimukset

Palvelin

Palvelinohjelmisto backend

Tulee olla Apache tai mahdollisesti jokin muu PHP -ohjelmointikieltä tukeva. Ratkaisuvaihtoehdosta riippuen palvelimella voidaan myös tarvita Java-tukea.

Palvelinohjelmisto frontend

Tarkoitus on, että asiakkaan puolen palvelinohjelmisto on perustasoa internetsivuille. PHP-tuki on kuitenkin tarpeellinen.

Ohjelmointikieli

Ohjelmisto toimii PHP-ohjelmointikielen versiolla 5.3.8. tai uudemmalla. Backendissä voidaan mahdollisesti käyttää myös Javaa ratkaisuvaihtoehdosta riippuen. Javascript ja sen erinäiset kirjastot ovat tarpeellisia frontend-puolella.

Tietokantapalvelin

Järjestelmän tietokanta toteutetaan MySQL-versiolla 5.5.18 tai uudemmalla.

Asiakas

Käyttöjärjestelmä

Asiakkaan käyttöjärjestelmällä ei ole juurikaan merkitystä, kunhan selain on jokin tuetuista ja se on päivitetty ajantasalle.

Selain (työpöytä)

Järjestelmän frontendissä käytetään ainakin Javascriptiä, joten siihen täytyy selaimesta tuki löytyä. Kaikista nykyajan selaimista löytyy JS-tuki. Järjestelmä toteutetaan ja testataan seuraaville selaimille:

Google Chrome versio 8 tai uudempi
Mozilla Firefox 8 tai uudempi
Internet Explorer 8 tai uudempi
Safari 5 tai uudempi
Opera 10 tai uudempi
SeaMonkey 5 tai uudempi

Selain (mobiili)

Mobiiliselaimestakin on löydyttävä Javascript tuki. Järjestelmä toteutetaan ja testataan seuraaville mobiiliselaimille:

Android 4.0 tai uudempi
Safari 5 tai uudempi
Nokia Browser 7.1 tai uudempi
Opera Mobile 11.5 tai uudempi

Interne-tyhteys

Internet-yhteyden vähimmäisnopeus järjestelmän frontendin käyttämiseen on 512 kbit/s. Suositeltu nopeus on >2Mbps eli tavallinen laajakaistayhteys.

Laadulliset tavoitteet

Käyttöliittymäratkaisut pyritään toteuttamaan käyttäjäystävällisesti ja loogisesti. Järjestelmä toteutetaan alustavasti suomen kielellä, mutta käyttöliittymä on tehtävä siten, että kielen muuttaminen tapahtuu esimerkiksi erillisillä tiedostoilla eikä itse runko-ohjelmaan tarvitse kajota. Järjestelmän lähdekoodi dokumentoidaan ja kommentoidaan kattavasti jatkokehityksen helpottamiseksi. Järjestelmälle suoritetaan kehitysvaiheen aikana yksikkö-, integrointi- ja järjestelmätestaus.

Vaatimusmäärittelyvaiheessa kerätään myös vaatimuksia, joita ei välttämättä toteuteta tässä versiossa. Ne ovat ohjelmiston jatkokehittäjiä varten. Järjestelmän on pyrittävä olemaan vikasietoinen. Järjestelmän laatua varmistetaan katselmoinneilla. Katselmointeja tehdään vaatimusmäärittelydokumenttiin. Kaikissa järjestelmän syötekentissä on noudatettava sopivaa syötetiedon tarkistusta. Järjestelmän tulee olla helposti siirrettävissä eri palvelimelle. Järjestelmä tulee olla helposti poistettavissa ainakin frontend-palvelimelta.

Käyttäjävaatimukset

Tässä osiossa esitellään ajanvarausjärjestelmän käyttäjäryhmät sekä näiden ryhmien vaatimukset.

Käyttäjäryhmät

Käyttäjäryhmiä on kolme. Ajanvarausjärjestelmää käyttää asiakas, joka haluaa varata palveluita. Järjestelmää saattaa käyttää myös työntekijä, joka merkitsee omat työtuntinsa, joihin varauksia voi tehdä. Kolmas ryhmä on ylläpitäjä, joka pystyy esimerkiksi lisäämään, poistamaan tai muokkaamaan palveluita ja työntekijöitä. Riippuen järjestelmän käyttöympäristöstä, resurssina toimiva ihminen, esimerkiksi hieroja voi olla toisessa ympäristössä vuokrattava huone. Tällöin työntekijän roolia ei välttämättä tarvita. Nämä kaikki ryhmät ovat frontendin käyttäjiä. Backendin eli itse runko-ohjelman ylläpitäjät ovat erikseen.

Molemmat ylläpidolliset ryhmät, työntekijä ja ylläpitäjä jakavat, vaatimuksia keskenään. Järjestelmässä voidaan määritellä ryhmille annettavat oikeudet.

Toiminnalliset vaatimukset

Vaatimuksille on jaettu prioriteetti seuraavasti:

Tärkeä: olennainen osa käyttökelpoista järjestelmää.

Keskinkertainen: Parantaa ohjelmiston käyttökelpoisuutta huomattavasti. Ohjelmisto kuitenkin toimii ilman sitä. Suurin osa keskinkertaisista vaatimuksista toteutetaan ensimmäisessä tuotteen versiossa.

Matala: Tuo lisäarvoa jo käyttökelpoiseen järjestelmään. Se voidaan toteuttaa mahdollisesti tulevien iteraatioiden aikana.

Vaatimusten numerointi

Vaatimukset on numeroitu ja niihin on liitetty kirjaintunnus ryhmästä riippuen. Ryhmien tunnukset ovat:

Asiakas, **A**
Työntekijä, **T**
Ylläpitäjä, **Y**

Asiakas

Tämä loppukäyttäjäryhmä koostuu suurimmalta osin kotikäyttäjistä, jotka tekevät ajanvarauksia eri palveluihin eri palveluntarjoajilta.

Prioriteetti: tärkeä
A1. Asiakas voi valita haluamansa resurssin.
A2. Asiakas voi valita haluamansa palvelun/palvelut.
A3. Asiakas voi valita haluamansa kellonajan ja päivämäärän varaukselle.
A4. Asiakas voi tarkastaa haluamansa palvelun hinnan.
A5. Asiakas voi perua varauksen.

Prioriteetti: keskinkertainen
A6. Asiakas voi rekisteröityä järjestelmään.
A7. Asiakas voi kirjautua järjestelmään.
A8. Asiakas voi muokata omia tietojaan.
A9. Asiakas voi poistaa itensä palvelusta halutessaan.
A10. Asiakas voi tilata salasanan rekisteröinnissä annettuun sähköpostiosoitteeseen.
A11. Asiakas voi jättää palautetta palvelusta.
A12. Asiakas voi lähettää viestin ylläpidolle tai työn tekijälle.
.
Prioriteetti: matala
A13. Asiakas voi tunnistautua luotettavaksi käyttäjäksi.
A14. Asiakas voi selailla käynti- ja palveluhistoriaansa.
A15. Asiakas voi halutessaan valita teksti- tai/ja sähköpostiviestimuistutuksen varatusta ajasta.

Työntekijä

Järjestelmässä voi olla käytössä työntekijä-ryhmä, jos ympäristö sen vaatii. Esimerkiksi parturikampaamossa voi olla yksi ylläpitäjä ja useita työntekijöitä, joilla on alhaisemmat käyttäjäoikeudet.

Prioriteetti: tärkeä

T1. Työntekijä voi merkata omat työaikansa, jonka perusteella varauksia voi tehdä.
T2. Työntekijä voi muokata omia tietojaan.
T3. Työntekijä voi liittää itsellensä palveluita.
T4. Työntekijä voi poistaa itseltään palveluita.

Prioriteetti: keskinkertainen

T5. Työntekijä voi tulostaa merkattujen työaikojen ja varausten perusteella työaikalistan.
T6. Työntekijä voi luoda työaikarungon.
T7. Työntekijä voi poistaa työaikarungon.
T8. Työntekijä voi muokata työaikarunkoa.

Prioriteetti: matala

T9. Työntekijä voi lisätä tauon työaikaansa.

Ylläpitäjä

Tällä ryhmällä on kaikista suurimmat oikeudet ajanvarausjärjestelmässä. Jossakin ympäristössä tämä voi olla ainut ryhmä asiakkaan lisäksi. Esimerkiksi liikuntahallin varauksissa ei välttämättä tarvita muuta ylläpidollista ryhmää.

Prioriteetti: tärkeä

Y1. Ylläpitäjä voi lisätä asiakkaalle tilin.
Y2. Ylläpitäjä voi poistaa asiakkaan.
Y3. Ylläpitäjä voi muokata asiakkaan tietoja.
Y4. Ylläpitäjä voi lisätä työntekijälle tilin.
Y5. Ylläpitäjä voi muokata työntekijän tietoja.
Y6. Ylläpitäjä voi poistaa työntekijän.
Y7. Ylläpitäjä voi lisätä palvelun.
Y8. Ylläpitäjä voi poistaa palvelun.

Y9. Ylläpitäjä voi muokata palvelun tietoja.

Y10. Ylläpitäjä voi liittää palvelun työntekijälle.

Y11. Ylläpitäjä voi poistaa palvelun työntekijältä.

Y12. Ylläpitäjä voi lisätä varauksen.

Y13. Ylläpitäjä voi perua varauksen.

Prioriteetti: keskinkertainen

Y14. Ylläpitäjä voi passivoida työntekijän tilin.

Y15. Ylläpitäjä voi liittää työntekijälle varattavia aikoja.

Prioriteetti: matala

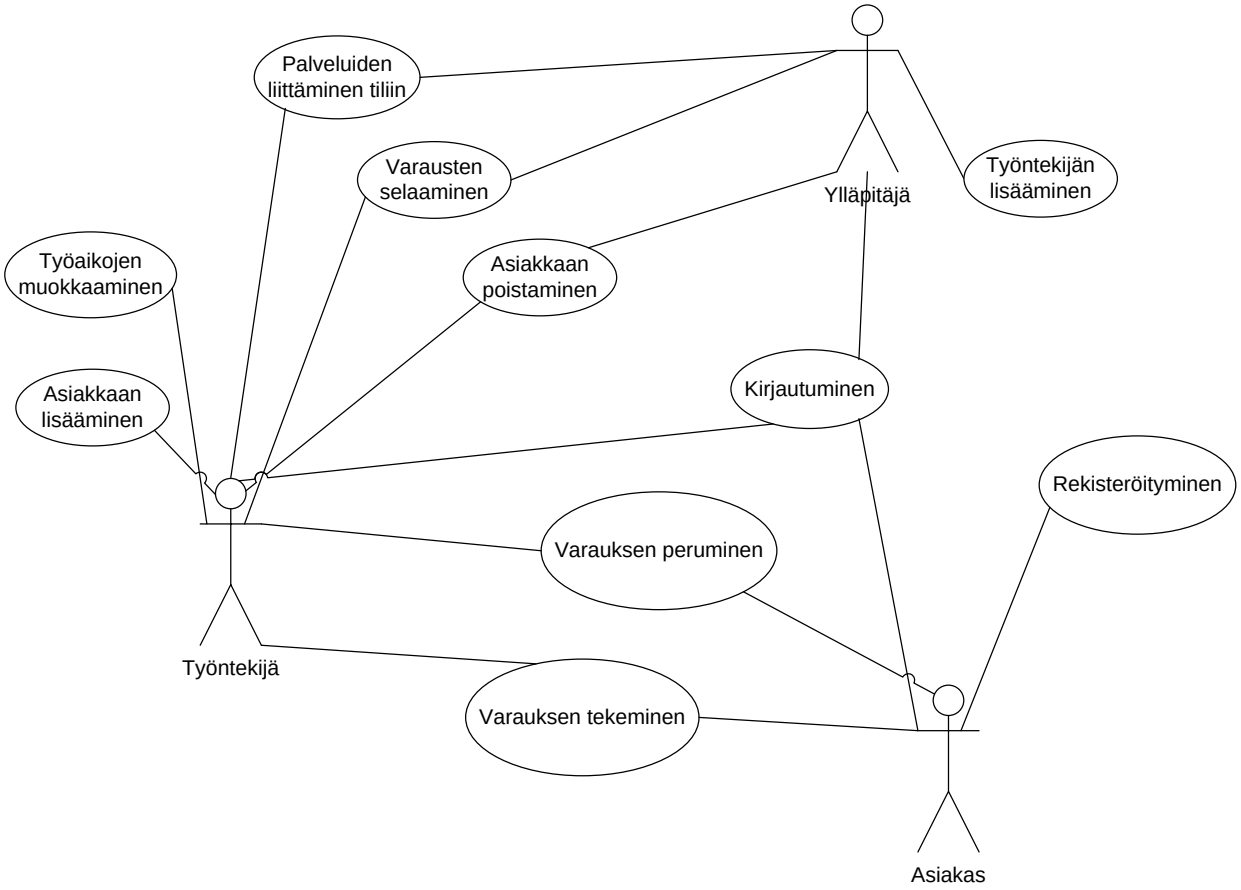
Y16. Ylläpitäjä voi lähettää järjestelmän kautta asiakkaille asiakaskirjeen.

Y17. Ylläpitäjä voi liittää asiakkaan tietoihin edellisten käyntien palvelut.

Y18. Ylläpitäjä voi liittää asiakastapahtumille lisätietoja.

Järjestelmän toiminnot

Käyttötapauskaavio



Käyttötapaukset

Käyttötapaus 1: Järjestelmään rekisteröityminen

Suorittaja: Asiakas

Käytettävyysvaatimukset: Rekisteröityminen tapahtuu yleisesti tunnetulla tavalla, jossa käyttäjä syöttää pyydetyt tiedot järjestelmään.

Alkuehdot: Käyttäjä on järjestelmän rekisteröitymissivulla.

Kuvaus: Käyttäjä syöttää kenttiin pyydetyt tiedot. Käyttäjä rekisteröityy palveluun. [Poikkeus: sähköpostiosoite on jo käytössä] [Poikkeus: syötetty tieto ei täytä ehtoja]
Poikkeukset: Sähköpostiosoite on jo käytössä: järjestelmä kertoo rekisteröitymisen epäonnistuneen jo käytössä olleen sähköpostiosoitteen takia ja ohjaa syöttämään toisen sähköpostiosoitteen. Syötetty tieto ei täytä ehtoja: järjestelmä kertoo rekisteröitymisen epäonnistuneen, koska joku/jotkut syötteistä eivät täytä tietojen vaatimuksia. Järjestelmää ohjaa syöttämään tiedot vaaditulla tavalla.
Loppuehdot: Käyttäjä on rekisteröitynyt järjestelmään.
Täyttää vaatimukset: A6

Käyttötapaus 2: Järjestelmään kirjautuminen

Suorittaja: Asiakas, Ylläpitäjä, Työntekijä
Käytettävyysvaatimukset: Kirjautuminen tapahtuu yleisesti tunnetulla tavalla, johon käyttäjät ovat tottuneet.
Alkuehdot: Käyttäjällä on käyttöoikeudet järjestelmään.
Kuvaus: Käyttäjä syöttää sähköpostiosoitteen ja salasanan. Käyttäjä kirjautuu järjestelmään. [Poikkeus: sähköpostiosoite tai salasana väärin] [Poikkeus: syötetty tieto sisältää ei-sallittuja merkkejä]
Poikkeukset: Sähköpostiosoite tai salasana väärin: järjestelmä kertoo kirjautumisen epäonnistuneen ja ohjaa syöttämään sähköpostiosoitteen ja salasanan uudelleen. Syötetty tieto sisältää ei-sallittuja merkkejä: järjestelmä kertoo kirjautumisen epäonnistuneen ja ohjaa syöttämään vain sallittuja merkkejä.
Loppuehdot: Käyttäjä on kirjautunut järjestelmään.
Täyttää vaatimukset: A7

Käyttötapaus 3: Ajan varaaminen / -asiakas

Suorittaja: Asiakas
Käytettävyysvaatimukset: Käyttäjä voi varata palvelun ja sen mahdollisen tuottajan tarpeidensa mukaan.

Alkuehdot: Käyttäjä on kirjautunut järjestelmään.
Kuvaus: Käyttäjä valitsee haluamansa palvelun, tuottajan ja ajankohdan.
Poikkeukset:
Loppuehdot: Varattu aika on tallennettu järjestelmään.
Täyttää vaatimukset: A6

Käyttötapaus 4: Ajan varaminen / ylläpitäjä, työntekijä

Suorittaja: Ylläpitäjä, Työntekijä
Käytettävyysvaatimukset: Käyttäjä voi varata asiakkaalle palvelun ja mahdollisen tuottajan asiakkaan tarpeiden mukaan.
Alkuehdot: Käyttäjä on kirjautunut järjestelmään.
Kuvaus: Käyttäjä valitsee asiakkaan, palvelun, mahdollisen tuottajan ja ajankohdan. [Poikkeus: uusi asiakas]. Käyttäjä tekee varauksen järjestelmään.
Poikkeukset: Uusi asiakas: käyttäjä siirtyy uuden asiakkaan lisäämiseen.
Loppuehdot: Varattu aika on tallennettu järjestelmään.
Täyttää vaatimukset: Y12, Y1

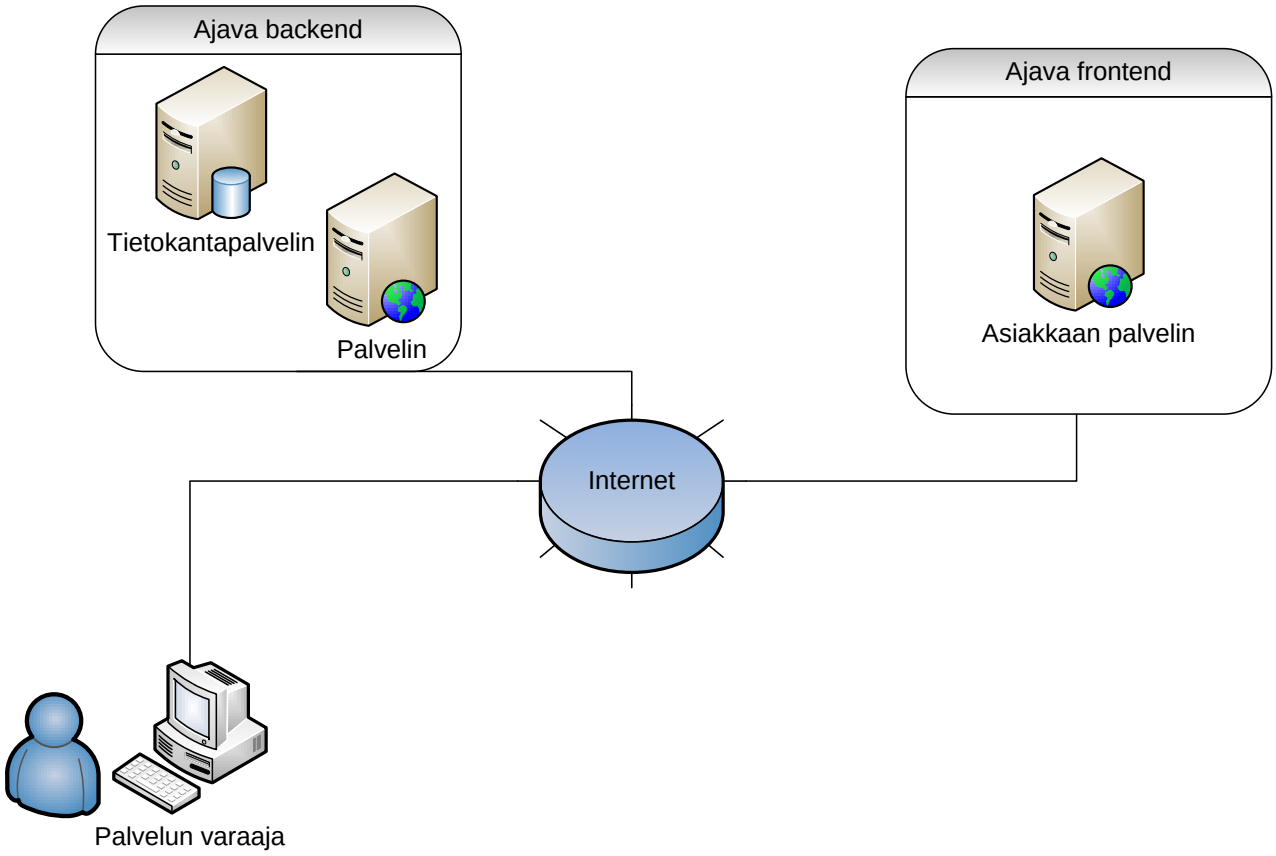
Käyttötapaus 5: Työajan lisääminen

Suorittaja: Työntekijä
Käytettävyysvaatimukset: Käyttäjä voi lisätä itselleen työaikoja viikoiksi eteenpäin joko samalla pohjalla tai vaihtelevasti.
Alkuehdot: Käyttäjä on kirjautunut järjestelmään.
Kuvaus: Käyttäjä valitsee työaikojen lisäämisen. Käyttäjä valitsee ajankohdan. Käyttäjä valitsee valmiin työaikapohjan tai lisää tunnit manuaalisesti. Käyttäjä tallentaa muutokset tai lisäykset.
Poikkeukset: -
Loppuehdot: Työajat on lisätty järjestelmään asiakkaiden nähtäväksi ja varattavaksi.
Täyttää vaatimukset: T1, T6, T8

Käyttötapaus 6: Varatun ajan peruminen

Suorittaja: Asiakas
Käytettävyysvaatimukset: Asiakas voi peruuttaa varaamansa ajan sopivan aikarajan puitteissa. Aikarajan voi määritellä järjestelmän ylläpitäjä.
Alkuehdot: Käyttäjä on kirjautunut järjestelmään.
Kuvaus: Käyttäjä valitsee ajan peruuttamisen. Käyttäjä syöttää varausvahvistuksessa saamansa peruutuskoodin kenttään. Käyttäjä peruttaa ajan. [Poikkeus: peruutuksen aikaraja ylittynyt] [Poikkeus: peruutuskoodi on virheellinen]
Poikkeukset: Peruutuksen aikaraja ylittynyt: Järjestelmä ilmoittaa, että aikaraja on ylittynyt, minkä jälkeen peruutusta ei voida enää itse suorittaa ja kehottaa käyttäjää ottamaan yhteyttä palveluntarjoajaan puhelimitse. Peruutuskoodi on virheellinen: järjestelmä ilmoittaa koodin olevan virheellinen ja syöttämään sen uudelleen.
Loppuehdot: Varattu aika on peruutettu ja varatun työn tekijälle lähtee ilmoitusviesti.
Täyttää vaatimukset: A5

Järjestelmäarkkitehtuuri



Sovellus toteutetaan pääpiirteittäin MVC-arkkitehtuuria noudatellen. Mikäli tuotteen runko tehdään PHP:lla, käytetään sen tekemisessä Codeigniter frameworkia. Kyseinen framework parantaa kielen MVC-ominaisuuksia ja tuo enemmän oliomaisia piirteitä ohjelmointiin.

Sovelluksen arkkitehtuuriratkaisut

Järjestelmän noudatellessa MVC-mallia ulospäin näkyvä eli view layer rakennettaan XHTML-, HTML- ja Javascript-kielillä. Välissä on PHP controller, joka välittää ja käsittelee käyttäjältä tulleet syötteet ja komennot. Model-kirjasto on myös toteutettu PHP:lla. Järjestelmän tiedot tallennettaan tietovarastoon, joka on MySql-tietokanta. Tietokanta ei ole samalla fyysisella palvelimella kuin PHP:lla toteutetut näkymä- ja ohjain-osat. Näkymä- ja ohjainkerrokset löytyvät aina asiakkaan omalta palvelimelta.

Elinkaari

Projektin elinkaari pitää sisällään määrittelyn, suunnittelun, toteutuksen, testauksen ja ylläpidon. Elinkaaren loppupäässä on mahdollinen järjestelmän käytöstä poistaminen.

Projektin työ / vaiheet

Määrittely:
Projektisuunnitelman laadinta
Vaatimusmäärittely
Suunnittelu:
Järjestelmän suunnittelu
Testaussuunnitelman luonti
Toteutus:
Varsinainen ohjelmointityö
Testaus:
Testaus testaussuunnitelman mukaan
Järjestelmän osien testaus
Ylläpito:
Tuotteen mahdolliset päivitykset
Korjaukset
Jatkokehitys

Lähteet

http://gs.statcounter.com/ - tilastot

http://www.opera.com/mobile/download/versions/ - opera mini

http://browser.nokia.com/update-browser.html - nokia

http://www.w3schools.com/browsers/browsers_explorer.asp - w3schools