SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

TYÖAJANSEURANTA-JÄRJESTELMÄ

Opinnäytetyö

TEKIJÄ/T: Juho Korhonen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Tietotekniikan koulutusohjelma	
Työn tekijä(t) Juho Korhonen	
Työn nimi Työajanseurantajärjestelmä	
Päiväys 17.3.2014	Sivumäärä/Liitteet 28/1
Ohjaaja(t) Lehtori Sami Lahti Lehtori Jussi Koistinen	
Toimeksiantaja/Yhteistyökumppani(t) Tamtron Solutions Oy	

Tiivistelmä

Tämän opinnäytetyön aiheena on työajanseurantajärjestelmä, joka tuo Tamtron Solutions Oy:lle nykyaikaisilla työkaluilla toteutetun tuotteen.

Vanha järjestelmä on toteutettu vanhemmilla tekniikoilla ja tekniikan osaajien löytäminen vie aikaa, joten Tamtron Solutions Oy haluaa korvata vanhan järjestelmän nykyaikaisella. Järjestelmän laajuuden takia opinnäytetyön pääaiheena on käyttäjille suunnattu järjestelmänhallintaan käytettävä työpöytäsovellus.

Järjestelmä on toteutettu vahvasti nojaten Microsoftin .NET Framework -ohjelmistokehykseen ja siihen pohjautuviin kirjastoihin. Tietokanta on toteutettu Entity Frameworkilla ja tietokantamoottorina toimii Microsoft SQL Server 2008. Tietoliikenneyhteydet on toteutettu käyttäen WCF-kirjastoa.

Opinnäytetyöraportin kirjoitusvaiheessa järjestelmä on vielä keskeneräinen ja järjestelmän odotetaan olevan myyntivalmis lähivuosina. Työajanseurantajärjestelmän dokumentit on luokiteltu salaisiksi eikä niitä siksi ole liitetty osaksi opinnäyteraporttia.

Avainsanat
.NET, C#, työajanseuranta, WCF

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Juho Korhonen			
Title of Thesis Working Time Attendance System			
Date	17 March 2014	Pages/Appendices	28/1
Supervisor(s) Mr. Sami Lahti, Lecturer Mr. Jussi Koistinen, Lecturer			
Client Organisation /Partners Tamtron Solutions Oy			

Abstract

The main subject of this thesis was a working time attendance system developed with state of art technology. This will bring a new product to Tamtron Solutions Oy's product range.

The old working time attendance system uses old technologies and it is increasingly hard to find people with expertise in these technologies. Due to how large the new system will be, this thesis was about the management software which customers will be using.

The system heavily depends on Microsoft's .NET Framework and its component libraries. The database opereates using Entity Framework and the database instance itself is operated with the Microsoft SQL Server 2008 software. The communication interface was developed with the WCF library.

As of writing of this thesis, the system is still in development and is excpected to be ready for sale in coming years. Most documents for this system are considered confidental and thus cannot be included in this thesis.

Keywords
.NET, C#, wroking time attendance, WCF

ESIPUHE

Tämä opinnäytetyö tehtiin syksyllä 2013 ja talvella 2014 Tamtron Solutions Oy:lle. Työn ohjaajana toimi lehtori Sami Lahti Savonia-ammattikorkeakoulusta. Haluan kiittää häntä sekä ohjauksesta että työn tarkastamisesta. Tämän lisäksi kiitokset Tamtron Solutions Oy:n henkilökunnalle tuesta ja neuvoista.

Espoossa 17.3.2014
Juho Korhonen

SISÄLTÖ

LYHENTEET JA MÄÄRITELMÄT

.NET

.NET Framework lyhennettynä. Opinnäytetyössä käytetty Mircrosoftin kehittämä ja ylläpitämä ohjelmistokehys. Tarkempaa tietoa löytyy .NET Framework -kappaleessa.

WCF

Lyhenne sanoista Windows Communication Foundation. Yhteysrajapintakirjasto, joka on osa .NET Framework -ohjelmistokehystä.

WPF

Lyhenne sanoista Windows Presentation Foundation. Käyttöliittymäkomponenttikirjasto, joka on multimediaominaisuuksiltaan WinForm-kirjastoa monipuolisempi. Tämäkin kirjasto on osa .NET Framework ohjelmistokehystä.

C#

Microsoftin .NET Framework -ohjelmistokehykseen kehittämä ohjelmointikieli.

Kirjasto

Yhteen kasattu paketti valmiita ohjelmisto-osia ja toimintoja.

Moduuli

Ohjelman osa, joka on vaihdettavissa ja/tai uudelleen käytettävissä toisessa projektissa.

WinForm

Lyhenne sanoista Windows Form. Viitataan .NET Frameworkin yleisesti käytettyyn käyttöliittymäkomponenttikirjastoon.

Työajanseurantajärjestelmä

Koko järjestelmä, joka tarvitaan työajanseurannan toimimiseen. Käsite sisältää tietokannan, tietoliikennerajapinnan sekä työpöytäsovelluksen.

Työpöytäsovellus

Sovellus, jota järjestelmän käyttäjät käyttävät työpäätteillään. Pääpainoisesti työpöytäsovellusta kehitetään järjestelmänylläpitäjiä ajatellen, mutta perustyöntekijöitä ei ole unohdettu heitä mahdollisesti koskevissa näkymissä.

1 JOHDANTO

Tämän opinnäytetyön aiheena on toteutettu työajanseurantajärjestelmä. Järjestelmän tarkoituksena on tuoda uudella tekniikalla kehitetty työajanseuranta uusille ja vanhoille asiakkaille.

Tamtron Solutions Oy haluaa luoda tällä hetkellä myynnissä olevan räätälöidyn työajanseurantajärjestelmän vierelle uudella tekniikalla kehitetyn pakettiratkaisun, jota voidaan tarjota pienemmällä räätälöinnillä asiakkaille ja näin laajentaa markkina-aluetta.

Projekti tulee aluksi kattamaan hyvin pienen osan nykyisen työajanseurantajärjestelmän ominaisuuksista. Järjestelmä on tarkoitus tehdä hyvin modulaariseksi, jotta työajanseurantajärjestelmää olisi helppo jatkokehittää kattavaksi järjestelmäksi. Tämän lisäksi modulaarisuus mahdollistaa ennalta määriteltyjen peruspakettien myynnin ja markkinoinnin.

2 KÄYTETYT TEKNIIKAT JA TYÖKALUT

Tässä luvussa on käyty läpi käytetyt tekniikat ja työkalut yleisellä tasolla. tekniikkoista ja työkaluistaa kerrotaan lyhyesti historiaa miksi niitä päätettiin käyttää sekä missä ja miten niitä käytetään.

2.1 Tekniikat

2.1.1 .NET Framework

.NET Framework on Microsoftin kehittämä ohjelmistokehys. Ensimmäinen versio .NET Frameworkistä ilmestyi vuoden 2002 alusta. Kirjoitushetkellä uusin .NET Framework -versio on 4.5.1, joka julkaistiin vuoden 2013 loppupuolella. (Microsoft 2013-10-22.)

.NET-ohjelmistokehyksessä on Javan tapaan automaattinen roskienkeräys eikä ohjelmoijan tarvitse itse huolehtia muistinhallinnasta. .NET Framework on laajasti tuettu eri ympäristöissä: sillä voi tehdä työpöytä-, internet-, tablet-tietokone- ja mobiilisovelluksia. .NET Framework -ohjelmat, webohjelmia lukuun ottamatta, toimivat vain ja ainoastaan Microsoft Windows -käyttöjärjestelmissä. Tämä ominaisuus onkin .NET Frameworkin suurin rajoitus. (Microsoft 2013.)

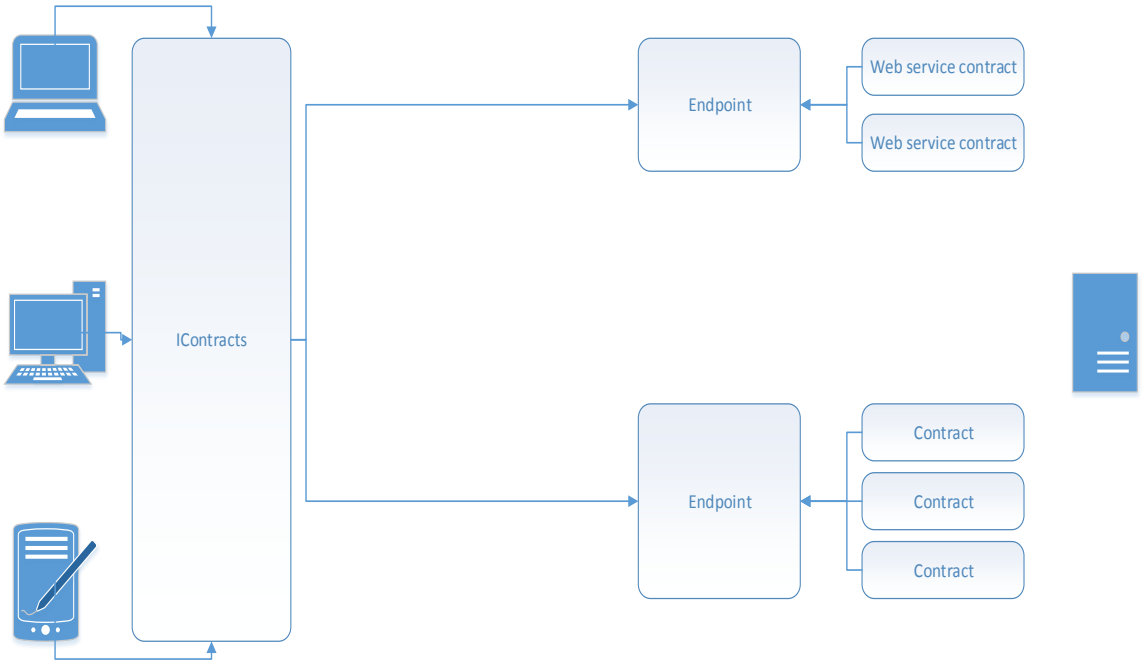
Hanselman (2010) suoritti epävirallisen Twitter-kyselyn, jossa hän tiedusteli, mitä .NET Frameworkin osaa ihmiset käyttävät. Hän julkaisi tulokset blogissaan. Näiden tulosten perusteella .NET Frameworkin yleisin käytetty osa on C#-ohjelmointikieli, joka kehitettiin nimenomaan .NET:ä varten. Seuraavaksi tulee ASP.NET, joka on .NET Frameworkin webkehys. ASP.NET-websivut vaativat sivua tarjoavalta koneelta tuen .NET Frameworkille, eli koneen käyttöjärjestelmä pitää olla Microsoft Windows, mutta ASP.NET-sivut eivät vaadi selaimilta tukea .NET Frameworkille. Kolmannella sijalla on WCF-tietoliikennekirjasto.

Tässä opinnäytetyössä tehtävä työajanseurantajärjestelmä nojautuu .NET Frameworkin sisältämiin tekniikoihin, ja tästä syystä järjestelmä on suunnattu Microsoft Windows -ympäristöön.

2.1.2 Windows Communication Foundation

Windows Communication Foundation (WCF) on .NET Frameworking osa, jota voidaan käyttää ohjelmien välisen yhteysrajapinnan toteutuksessa. Windows Communication Foundation julkaistiin ensimmäistä kertaa .NET Framework 3.0 -versiossa. Chappellin (2005-02) mukaan Windows Communication tunnettiin aluksi nimellä 'Indigo', mutta sittemmin nimi muutettiin Windows Communication Foundationiks.

Windows Communication Foundationilla voidaan toteuttaa palveluarkkitehtuurisia yhteysrajapintoja. Windows Communication Foundationilla voidaan myös toteuttaa yhteysrajapinta, johon ohjelmat voivat olla yhteydessä, vaikka ohjelma ei itse tukisi .NET Frameworkia. Kuviossa (KUVIO 1) on kuvattu esimerkkitapaus, jossa palvelimella on kaksi erillistä Endpoint-osoitetta ja molemmilla osoitteilla on omat sopimukset. Windows Foundation Communicationilla voidaan luoda sekä työpöytäsovellusten että internetsivujen yhteysrajapintoja. (Microsoft 2012-08-02.)



KUVIO 1. WCF-palveluesimerkki

Tässä opinnäytetyössä Windows Foundation Communicationilla on tehty yhteysrajapinta, jolla työpöytäsovellus on yhteydessä palvelimen tietokantaan. Yhteysrajapintaa valittaessa oli vaatimuksena, että rajapinnalla voisi toteuttaa seuraavat ominaisuudet:

- helposti toteutettava tietoturva
- tuki työpöytäsovelluksille
- tuki websivusovelluksille
- tuki Linuxiin.

Microsoft Communication Foundationissa yllä mainitut ominaisuudet löytyvät ja ovat helposti liitettävissä toimivaksi kokonaisuudeksi.

Yhteysrajapintaan on tehty hälytyssanomia, joita lähetetään sovelluksille tietokannan datan muutoksista. Tällä tavalla ei ohjelmistojen tarvitse erikseen varmistaa tietokannalta, onko dataan tullut muutoksia viime kysymästä.

2.1.3 Entity Framework

Entity Framework on olio-relaatio-kartoitukseen tarkoitettu ohjelmistokehys. Entity Framework sallii ohjelmoijalle suoran pääsyn dataan ilman erillisiä datan hakukoodeja, esim. SQL-lausekkeita. Entity Frameworkin tarkoitus on helpottaa näin datan käsittelyä poistamalla erillisen datan haku-, poisto- ja päivityskoodien tarpeen. (EntityFrameworkTutorial.net.)

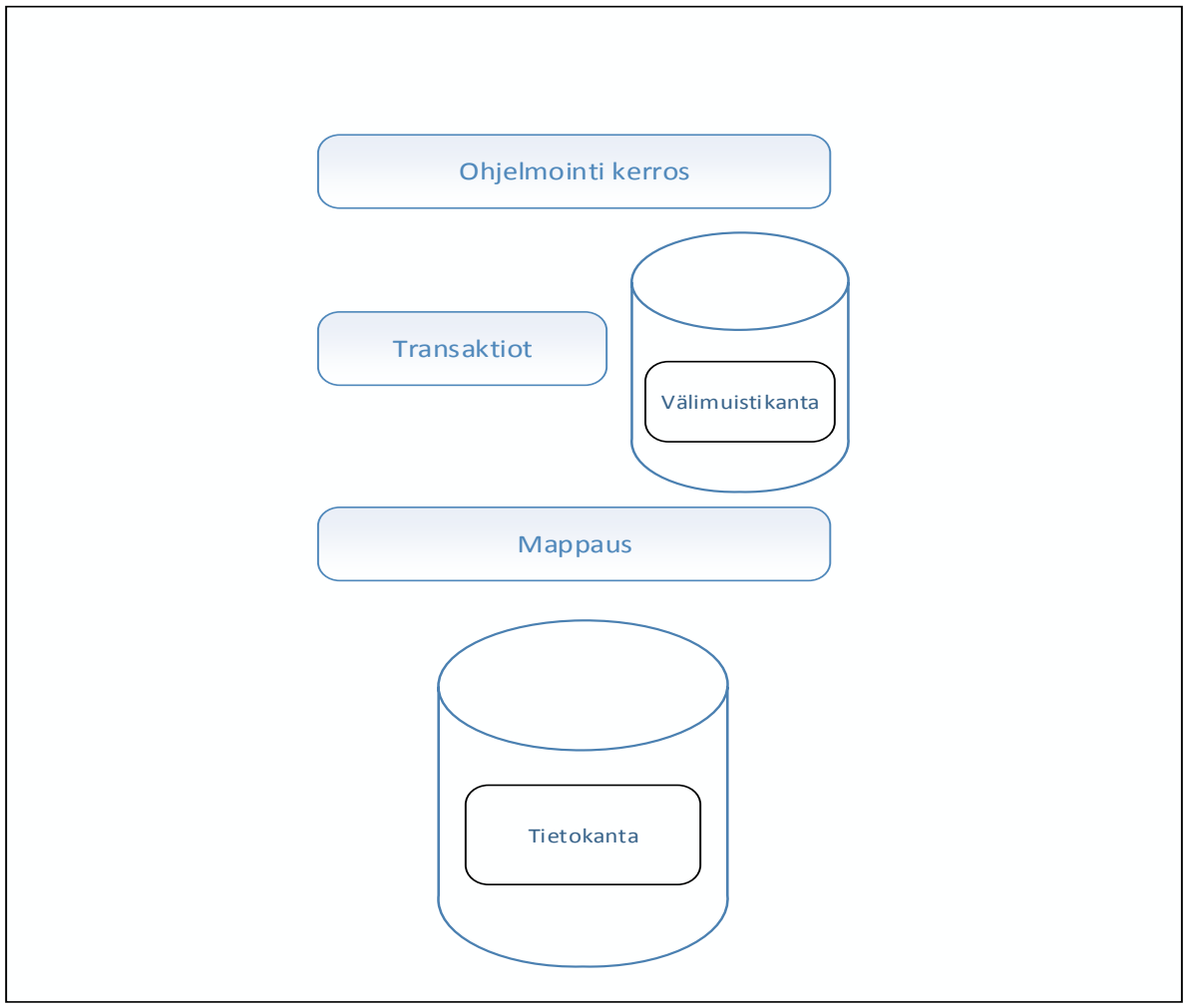
Entity Framework julkaistiin .NET Framework 3.5 Service Pack 1:ssä elokuussa 2008. Entity Framework sai uuden julkaisun .NET Framework 4.0:n kanssa huhtikuussa 2010. Uuden version oli tarkoitus parantaa edellisessä julkaisussa ilmenneitä ongelmia ja puutteita. (Microsoft 2009-04-11.)

Entity Frameworkin lähdekoodi on avointa koodia ja se on lisensoitu Apache 2.0 –lisenssillä (InfoWorld 2012-07-20). Entity Frameworkin koodi on saatavilla Microsoftin omistamalla CodePlex-sivustolla. Kyseisellä sivustolla voi myös käydä ilmoittamassa mahdollisista virheistä sekä tehdä ehdotuksia ominaisuuksista Entity Frameworkiin.

Entity Framework osaa luoda sille määritellystä oliorakenteesta tietokantarakenteen ja liitokset taulujen välille. Tämä prosessi on automaattinen, mutta sitä voidaan hallita erikseen ohjelmakoodillisesti ja tarvittaessa taulujen liitoksia voidaan lisätä tai purkaa.

Alla olevassa kuviossa (KUVIO 2) on kuvattu yksinkertaisesti, mistä kerroksista Entity Framework koostuu. Kaikki koodi Entity Frameworkissa kirjoitetaan ohjelmointikerrokseen.

Transaktiokerros hoitaa transaktioiden toteutumisen ja päivityksen. Välimuistikanta sisältää viimeksi haettua dataa hakujen nopeuttamiseksi. Mappauskerros hoitaa Entity SQL –rakenteen muuntamisen käytössä olevan tietokannan SQL-rakenteeseen.



KUVIO 2. Entity Frameworkin kerrosrakenne

Entity Framework valittiin tähän opinnäytetyöhön siksi, että se mahdollistaa datan käsittelyn ilman erillistä kokemusta tietokannoista ja datan käsittelystä tietokannoissa ja tietokantoihin. Näin ohjelmoija, jolla on hyvinkin vähän kokemusta itse tietokannoista, voi avustaa datan käsittelyrajapinnan työstämisessä.

Entity Frameworkin ohjelmointiosuus toteutetaan C#-koodilla, ja ohjelmistokehys itse vastaa kaikesta datankäsittelystä tietokantaan ja takaisin.

2.1.4 C#

C# (C Sharp) on oliopohjainen ohjelmointikieli. C# on Microsoftin kehittämä ohjelmointikieli, joka lainaa piirteitä mm. C++ -kielestä, Javasta sekä Delphistä. C# pääarkkitehtina toimi Anders Hejlsberg, joka aiemmin toimi Borland:lla kehittämässä Turbo Pascalia.

C# julkistettiin ensimmäistä kertaa heinäkuussa 2000 ammattilaiskehittäjien konferenssissa. C# tunnettiin kehityksen ajan koodinimellä "Cool". (Noorul 2012-09-27.)

C# valittiin tämän projektin ohjelmointikieleksi, koska kyseisellä kielellä on laaja tuki eriasteisille valmiille ja puolivalmiille rajapintaratkaisuille. Tämän lisäksi C# oli ollut jo pitkään Tamtronilla pääohjelmointikielenä. Opinnäytetyössä kaikki osat on toteutettu C#-ohjelmointikielellä:

- WCF toimii yhteysrajapintana.
- Windows Forms ja WPF muodostavat käyttöliittymän.
- Entity Framework toimii tietokantamoottorina.

2.1.5 Windows Forms

Windows Forms on .NET Framework 2.0:ssa julkaistu käyttöliittymäkomponenttikirjasto. Windows Forms muistuttaa kovasti Microsoftin aiempaa C++:lla tekemää Microsoft Foundation Class –kirjastoa. (Microsoft 2003-04.)

WinFormit rakentuvat suoraan Windows APIn päälle ja Windows itse päättää, kuinka komponentit piirretään näytölle. Tästä johtuen WinForm-sovellukset ottavat käyttäjän käyttöjärjestelmän teema-asetukset käyttöönsä, ellei ohjelmaa ole ohjelmoitu toimimaan toisin. Jokainen WinForm-komponentti on olio, jolla on paljon esimääriteltyjä ominaisuuksia ja metodeja (Janssen).

Windows Form -sovelluksia kehitetään pääsääntöisesti graafisella käyttöliittymällä. Microsoftin omassa Visual Studio -kehitysympäristössä on täysi tuki graafiseen käyttöliittymäsuunnitteluun käyttäessä niin Windows Form kuin WPF-komponentteja.

Opinnäytetyö päätettiin toteuttaa Windows Form -sovelluksena, koska WinForm-sovellukset ovat yhteensopivia aiempien Windows-käyttöjärjestelmien kanssa, kunhan käyttöjärjestelmä

vain tukee oikeaa .NET Framework -versiota. Lisäksi WinFormit tarjosivat paljon toiminnallisuutta valmiissa paketissa, mitä järjestelmä kaipasi. Windows Form -sovelluksiin on myös helposti integroitavissa Microsoft SQL tietokanta ja siihen Microsoftin tarjoamat työkalut.

2.1.6 Windows Presentation Foundation

Windows Presentation Foundation on .NET Framework 3.0:ssa julkaistu käyttöliittymäkomponenttikirjasto. WPF:n on tarkoitus tarjota käyttöliittymäkomponentteja, jotka ovat graafisesti vaativimpia, kuin Windows Form -komponentit. WPF-komponentit tukevat animointia, 3D-kuvaa, erikoistehosteita sekä ääntä. (Microsoft.)

Windows Presentation Foundation -komponenttien runko rakentuu XAML-koodilla. XAML on lyhenne sanoista Extensible Application Markup Language (Laajennettava ohjelmistomerkintäkieli). XAML perustuu XML-merkintäkieleen.

Guthrie (2006-12-04) kertoo että, Silverlight on koodinimeltään WPF/E ja pohjautuu WPF-tekniikkaan, mutta on tarkoitettu toimimaan selainympäristössä. Silverlight käyttää muun muassa XAML:ia ja sisältää samoja graafisia komponentteja kuin WPF-kirjasto (Microsoft).

Opinnäytetyössä päätettiin käyttää WPF-komponentteja, koska verrattuna WinForm-komponentteihin WPF-komponenttien graafinen suorituskyky ja muunneltavuus ovat paljon paremmat. Näistä eduista huolimatta, tekijöiden vähäisen WPF-kokemuksen takia, ohjelmanrunko päätettiin toteuttaa Windows Forms -tekniikalla.

WPF-toteutusta löytyy käyttäjäläheisistä moduuleista, joiden visuaalinen ulkomuoto on liian vaativa WinForm-komponenteille. Näiden moduulien graafiseen ulkoasuun on kiinnitetty erityistä huomiota, jotta tieto olisi mahdollisimman helposti saatavilla ilman, että ulkoasu häiritsisi käyttöä laajalla väriskaalalla.

2.2 Työkalut

2.2.1 Visual Studio 2012

Visual Studio 2012 on Microsoftin julkaisema kehitysympäristö. Visual Studion ensimmäinen julkaisu tapahtui vuonna 1995 ja Thurrottin (2012-05-18) mukaan sen sisäinen koodinimi oli Boston viitaten Bostonin kaupunkiin. Visual Studio 2012 on yhdeksäs Visual Studio -tuoteperheen julkaisu. Somasegar (2012-08-01) kertoo, kuinka Visual Studio 2012 julkaistiin syyskuussa 2012 samaan aikaan, kun Windows 8 ja Windows Server 2012 julkaistiin.

Visual Studio 2012:n valinta ohjelmisto projektissa oli helppo: työajanseurantajärjestelmää rakennetaan Microsoftin valmistamilla tekniikoilla, joten Microsoftin kehitysympäristö oli helppo valinta.

Opinnäytetyössä käytetään Visual Studio 2012 Premium -versiota. Premium-versio tarjoaa lisäominaisuuksia, joita Visual Studio Professional –versiossa ei ole (MicroWay). Näitä lisäominaisuuksia käytetään mm. etsiessä koodista pullonkauloja ja erityisesti konetta rasittavia koodipätkiä.

2.2.2 ReSharper

ReSharper on Visual Studioon liitettävä avustusohjelma, jonka tarkoitus on nopeuttaa koodin kirjoittamista, vähentää virheitä, yhtenäistää koodin ulkoasua sekä optimoida koodia. Esimerkkinä tästä voidaan kertoa käyttämättömien viittausten korostus (KUVIO 3): ReSharper korostaa käyttämättömät viittaukset kirjoittamalla nämä haaleammilla väreillä. ReSharper julkaistiin ensimmäistä kertaa 2004 heinäkuussa. Opinnäytetyössä käytetty ReSharper ohjelmisto on ReSharper 8.0, joka on julkaistu 2013 kesäkuussa. (JetBrains 2013.)

```
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
```

KUVIO 3. Käyttämättömien viittausten korostus

ReSharperiin on tehty Visual Studio 2012:een verrattuna paljon eriasteisia tarkistuksia, jotka tutkivat ja lukevat koodia heti sitä kirjoittaessa. Tällöin virheet ja mahdolliset optimointia kaipaavat koodit havaitaan heti koodin kirjoitusvaiheessa, eikä vasta kääntövaiheessa (Jgauffin 2013-02-11).

Opinnäytetyössä päätettiin käyttää ReSharperia sen tuoman ajallisen edun takia. ReSharper avustaa koodin yhdenmukaisuudessa, vaikka ohjelmaa tekisi useampi henkilö. ReSharper osaa myös luoda nopeasti puuttuvat olio- sekä metodimäärittelyt kesken koodin kirjoittamisen.

3 TYÖAJANSEURANTAJÄRJESTELMÄ

3.1 Käyttöliittymä

Käyttöliittymä on asia, jonka käyttäjä suoraan näkee ja kaikki toiminto tapahtuu sen kautta. Tästä syystä hyvä käyttöliittymä katsottiin yhdeksi ohjelman tärkeimmäksi osaksi ja sen saavuttamiseksi asetettiin kolme perusedellytystä:

- Helppokäyttöisyys
    - Ulkoasun tehtävä on mahdollistaa ohjelman mahdollisimman helppo käytettävyys.
    - Tarvittavan tiedon pitää olla saatavilla helposti eikä sitä pidä joutua hakemaan useasta eri paikasta.
- Selkeä ulkoasu
    - Ohjelman ulkoasu ei saa olla täytetty kaikenlaisilla painikkeilla tai muilla vastaavilla ohjauskomponenteilla.
    - Tiedonasettelu pitää olla selkeää, eikä tarvittava tieto saa olla puutteellista.
- Yhteneväisyys
    - Näkymät ovat yhdenmukaisia, eikä näkymien toimintatavoissa saa olla suuria eroja.
    - Näkymien ulkoasu muistuttaa toinen toistaan, eikä värimaailma muutu huomattavasti näkymien välillä.

3.1.1 Henkilöylläpitönäkymä

Henkilöylläpitonäkymää käytetään käyttäjien hallintaan, lisäykseen sekä poistoon. Tällä näkymällä voidaan määritellä henkilön perustiedot ja henkilön ryhmäasetukset sekä se mihin vuoroon henkilö kuuluu. Tätä näkymää käytetään myös, kun henkilölle luodaan käyttäjätunnus, jolla käyttäjä kirjautuu järjestelmään. Tämän takia käyttäjäylläpitonäkymä on perusosa työpöytäsovellusta ja on erittäin käytetty näkymä ohjelmassa.

Henkilönäkymää suunniteltaessa otettiin huomioon näkymän yleinen käyttö. Tästä huolimatta usea käyttökohde poikkeaa vahvasti toisistaan, vaikka käyttökohteet ovat vahvasti liitoksissa

toisiinsa. Tästä loistavana esimerkkinä voidaan mainita henkilönlisäys: Aluksi luodaan henkilö järjestelmään. Tämän jälkeen henkilölle annetaan kulkukortti ja oikeudet.

Tästä tilanteesta voidaan todeta, että henkilön ja henkilönkulkukortin lisäys ovat vahvasti yhdessä ja tiedon pitää olla helposti syötettävissä. Toisaalta joissain tapauksissa kulkukortti voi olla kierrossa monella ihmisellä lyhyin aikavälein ja tieto, kuka on kortin nykyinen omistaja, voi olla tarpeellinen. Voi myös olla, että kulkukortit luodaan lyhyille aikaväleille turvallisuussyistä ja halutaan tieto korteista, joiden kulkuoikeudet ovat vanhenemassa.

3.1.2 Työajanseurantanäkymä

Työajanseurantanäkymästä voidaan nähdä työntekijöiden toteutuneet työpäivät. Työajanseurantaan on tarkoitus sisällyttää sekä tarkastelu- että hyväksyntätoiminto. Työntekijä itse voisi tarkastella omia työtuntejaan ja todeta ne oikeiksi, minkä jälkeen työnjohto hyväksyy tarkastetut tunnit.

Koska työajanseurantanäkymää käyttää sekä työntekijät että työnjohto, on työajanseurantanäkymä suunniteltu mahdollisimman helppokäyttöiseksi ja informatiiviseksi. Työajanseurantanäkymä on toteutettu WPF-komponenteilla, koska niillä saadaan tieto korostettua helpommin kuin WinForm-komponenteilla.

Näkymää suunnitellessa piti huomioida, että yrityksissä ollaan kiinnostuneita eri asioista ja tietyt asiat eivät ole tärkeitä, kun taas toisessa yrityksessä voi olla päinvastainen tilanne. Hyväksi esimerkiksi voisi esittää työvuoron työajan ylittäneet päivät:

- Yrityksessä A työajanylitykset ovat merkityksettömiä ja ne leikataan yksinkertaisesti pois palkanlaskennassa. Mitään korostusta ei haluta, koska se veisi huomiota pois oikeasti tärkeistä tiedoista.
- Yrityksessä B työajanylitykset tulkitaan ylitöiksi ja niistä maksetaan ylityökorvausta. Mutta ylityön teko on kiellettyä ilman työnantajan lupaa, ja tästä syystä kaikki ylityöpäivät halutaan korostaa näkyvyyden lisäämiseksi.

Kyseisessä tilanteessa kaksi eri yritystä haluaa juuri päinvastaisen korostuksen ylityöpäiville, joten ylityön korostus pitää olla säädettävissä. Ylityöpäivät ovat vain yksi esimerkki ja korostustapauksia on lukuisia, joten tarvitaan toiminnallisuutta, jolla voidaan korostaa eri tapauksia.

Työajanseurantanäkymään ei tästä syystä suunniteltu toiminnallisuutta, jolla eri tapausten korostusta voitaisiin säätää käyttäjälle sopivaksi. Tätä toiminnallisuutta on myös helppo laajentaa sisältämään yhä useampi korostustoiminto.

3.1.3 Työvuoronäkymä

Työvuoronäkymällä luodaan ja hallitaan työvuoroja ja niissä kertyviä lisiä. Työvuorot voivat sisältää taukoja sekä lisiä. Työvuoronäkymän tavoitteena oli luoda helppo ja selkeä tapa luoda hyvinkin yksityiskohtaisia työvuoroja. Työvuoroille voidaan muun muassa määritellä eri päiville eri lisiä ja näille lisille voidaan määritellä yksityiskohtaisesti niiden alkamis- ja päättymisajankohta. Lisäksi lisille voidaan määritellä vähimmäisvaatimuksia ja määrätä lisän kertymiselle katto.

Työvuoronäkymän toteutusta ja suunnittelua haittasi sen tarvitseman tiedon suuri määrä. Työvuorot sisältävät tietoja sallituista lisistä, työpäivien kestoista sekä erikoispäivistä. Näiden lisäksi järjestelmä tulkitsee henkilöiden olevan tietyssä tilassa, esim. sisällä tai ulkona. Nämä tilat vaikuttavat myös lisien kertymiseen: järjestelmään voi määritellä, mitkä lisät kertyvät missäkin tiloissa.

Nämä kaikki sisältävät hyvinkin yksityiskohtaista tietoa työpäivän rakenteesta. Tästä syystä työvuoronäkymä on hyvin tietopohjainen näkymä ja tietoa on jaettu useaan eri välilehteen tietomäärän keventämiseksi. Täynnä tietoa oleva sivu on hämäävä ja vaikealukuinen etenkin kokemattomalle käyttäjälle. Tiedon suuri määrä myös hankaloittaa näkymän käyttöä ja oikean tiedon löytäminen on vaikeaa.

3.2 Työpöytäsovelluksessa käytetyt tekniikat

3.2.1 Windows Forms

Itse opinnäytetyönä tehty ohjelma on Windows Forms -sovellus ja suurin osa näkymistä on Windows Form -näkymiä. Aikaisemman Windows Forms -kokemuksen takia päädyttiin käyttämään tätä tekniikkaa peruspohjana käyttöliittymässä.

Windows Forms -kirjasto mahdollisti tapahtumapohjaisen toimintalogiikan, jolla hallitaan eri näkymien välistä tiedon yhteneväisyyttä. Kun yhden näkymän tieto muuttuu, saavat kaikki näkymät tiedon tästä muutoksesta. Tällä tavalla tieto näkymien välillä on synkronoitu reaaliajassa ja vältytään tilanteilta, joissa jollain näkymällä näytettäisiin vanhaa tietoa.

Aluksi tehtiin Windows Form -näkymät valmiiksi. Tämän jälkeen pohdittiin, mitkä näkymistä tarvitsisivat tarkempaa graafista ulkoasua ollakseen helppokäyttöisiä sekä selkeäkäyttöisiä. Useimmat näkymät olivat riittävän selkeitä ja helppokäyttöisiä Windows Form -tekniikalla tehtynä, mutta jotkut näkymät olivat Windows Form -tekniikalla joko epäselkeitä tai hankalia käytettäviä. Tämän takia kyseiset näkymät päätettiin tehdä uusiksi WPF-tekniikalla.

3.2.2 WPF

Näkymiä suunnitellessa Windows Form -komponentit osoittautuivat melko huonoksi vaihtoehdoksi, kun graafista toimintakykyä tarvittiin enemmän, vaikka Windows Form -komponenteilla oli paljon teknillistä toteutusta ja niitä oli helppo käyttää sellaisinaan. Tosin normaalista poikkeavissa käyttökohteissa Windows Form -komponenteissa alkoi ilmetä rajoitteita ja ongelmia. Esimerkkinä voidaan mainita muun muassa piirtonopeus- ja ruudunpäivitysongelmat.

Näistä johtuen tutkittiin vaihtoehtoisia tekniikoita toteuttamaan graafisesti vaativimpia näkymiä. Vaihtoehtoja läpikäydessä löytyi Windows Presentation Foundation, mikä jakaa monia ominaisuuksiaan Silverlightin kanssa.

WPF-komponentit tukevat laajennuksia ja niiden perustoimintaa voi muokata vapaammin kuin WinForm-komponentteja. Tämä vapaus tosin tekee WPF-komponenteista paljon monimutkaisempia kuin vastaavista WinForm-komponenteista. Siinä missä WinForm-komponentin DataGrid sisältää jo itsestään lajittelun, tässä opinnäytetyössä WPF-DataGridin lajittelu vaati noin 100 riviä koodia.

3.2.3 WCF

Työpöytäsovellus käyttää WCF-tekniikkaa tiedonsiirtoon tietokannan ja sovelluksen välillä. WCF-yhteys ei ole jatkuvasti auki, vaan se avataan tarvittaessa ja yhteyden tilaa tarkkaillaan mahdollisten virheiden havaitsemiseksi.

Työpöytäsovellus käyttää WCF-moduulia joka vuorostaan käyttää työajanseurantajärjestelmää varten tehtyä WCF-kirjastoa. Tällä saavutetaan tilanne, jossa tietoliikenneyhteys voidaan eristää käyttöliittymäkoodista ja tietoliikenneyhteys voidaan säikeistää.

WCF tarjoaa tietoturvaratkaisuna käyttäjän todentamiseen useita eri ratkaisuja mm. RS-A ja sertifikaattitodentamisen (Microsoft 2012-08-02). Eri todentamismenetelmiä ja niiden toteuttamistapoja on tutkittu lisäturvaksi opinnäytetyön suunnitteluvaiheessa, mutta niitä ei ole tätä opinnäytetyöraporttia kirjoitettaessa vielä toteutettu.

3.2.4 Tietokanta

Tietokantavalinta oli projektin alussa vielä avoin vaikka kehityksessä käytettiin Microsoft SQL Serveriä. Projektin edetessä Microsoft SQL Server on vahvistanut asemaansa tietokantavalintana, koska sen tiedonkeruu- ja raportointitoiminnallisuudet on helppo liittää kehitettävään työpöytäsovellukseen.

Tietokanta on relaatiokanta ja se on suurimmaksi osaksi Entity Frameworkin luoma rakenne. Kaikkia relaatioita ja poikkeustapauksia Entity Framework ei tähän järjestelmään osannut tehdä, vaan Entity Frameworkin alustukseen on lisätty poikkeustapauksia varten koodia, jotta poikkeustapausten relaatiot tulisivat oikein.

Entity Framework on mahdollistanut vähäisen tietokantakoodin, joka taas on nopeuttanut järjestelmän kehitysprosessia, kun tietokantataulujen muutokset vievät erittäin vähän aikaa. Tietokannan tiedon oikeellisuuden varmistaminen on ollut kohtalaisen haastavaa, koska tietokantaan kirjoittavat monet eri ohjelmat, usealta eri laitteelta ja vielä mahdollisesti eri tietoliikenneyhteyksillä.

Vähäisen tietokantaohjelmoinnin kokemuksen takia tiedon oikeellisuuden ylläpitäminen on osoittautunut odotettua monimutkaisemmaksi. Tiedon kirjoitus- ja päivityskomentoihin on jouduttu tekemään eriasteisia lukkoja ja tarkistuksia tiedon eheyden ja oikeellisuuden ylläpitämiseksi. Tiedon oikeellisuuden puute aiheuttaisi muun muassa tiedon menetyksiä, tiedon monistautumista ja pahimmassa tapauksessa virheellisiä arvoja ja järjestelmän lukkiutumista.

4 TESTAUS

Järjestelmän käytön kannalta on tärkeää, että se toimii katkotta koko työpäivän ajan. Myös työpäivän ajankohta vaihtelee yrityksien välillä. Joissakin tapauksissa työpäivä voi myös jatkua hetkinä, jolloin välitöntä tukea ei voida taata tai antaa. Näistä syistä johtuen on äärimmäisen tärkeää, että järjestelmä osaa toipua virhetilanteista, vaikka niitä tapahtuisi. Järjestelmän testauksessa kiinnitetäänkin paljon huomiota järjestelmän vakauteen.

Jokainen ohjelmoija joutuu suorittamaan yksikkötestausta omalle ohjelmakoodilleen. Tämän jälkeen ohjelmoijien ohjelmakoodit integroidaan keskenään ja molemmat ohjelmoijat suorittavat ohjelmistolle lyhyen testauksen. Kun virheiden määrä on minimoitu, testaa järjestelmää projektin ulkopuolinen henkilö.

Järjestelmän testausta edistetään myös käyttämällä järjestelmän osia niin paljon, kuin mahdollista uusia osia testatessa. Esimerkkinä voidaan mainita työpäivän työajanlaskenta:

- Aluksi järjestelmään tehdään leimaus leimauspäätteellä.
- Leimauksen tietoja muokataan halutuiksi työpöytäsovelluksella.
- Laskettuja työpäiviä tarkkaillaan työpöytäsovelluksella.

Järjestelmän eri osat tulevat näin käyttöön kehityksen aikana, jolloin käytettävyyteen voidaan puuttua erittäin aikaisessa vaiheessa kehitystä.

Järjestelmälle suoritetaan yrityksen sisäinen testi, jossa järjestelmää käytetään varsinaisen työajanseurantajärjestelmän rinnalla, jolloin järjestelmälle voidaan suorittaa käyttöönotto-, vakaus- sekä käytönhelppoustestit. Tällöin järjestelmää on testattu sekä käytetty järjestelmän tulevassa käyttöympäristössä ja käyttötarkoituksessa. Tämä testi on päätetty alkamaan tämän opinnäytetyön jälkeen eikä tarkempaa kuvausta testistä tai sen tuloksista voida siis kertoa.

5 TYÖN ARVIOINTI

Tavoitteena opinnäytetyössä oli saada Tamtron Solutions Oy:lle valmiiksi testiversio sisäiseen testaukseen. Ensimmäistä testiversiota odotettiin joulukuun loppupuolelle, mutta muut projektit söivät työajanseurantajärjestelmään varattua aikaa ja aikataulu siirtyi n. 1,5 kuukautta eteenpäin.

Järjestelmässä käytettiin myös paljon kohtalaisen uusia tekniikoita, joita opinnäytetyön tekijä ei ollut ennen käyttänyt. Tämän takia opinnäytetyön tekijältä kului normaalia enemmän aikaa näkymiä toteuttaessaan.

Järjestelmä on hyvällä alulla ja perusmoduulit ovat melko valmiita, mutta vaativat vielä paljon testejä. Tietokannan haku ja kirjoitusfunktiot ovat vielä kesken, sillä niitä on toteutettu vain tarpeen mukaan.

Opinnäytetyö on ollut erittäin haastava ja on tarjonnut mainion mahdollisuuden toteuttaa ja suunnitella ammattimaista järjestelmää nykyaikaisilla ohjelmistoilla ja työkaluilla. Järjestelmän kehitys tarjosi paljon tilaisuuksia käyttää jo opittuja taitoja ja perehtyä aiemmin tuntemattomiin tekniikoihin.

Opinnäytetyönä työstetty järjestelmä pyritään saamaan valmiiksi myyntiin lähivuosina. Toistaiseksi järjestelmässä on vielä työstettävää useassa moduulissa ja paljon testausta suoritettavana, mutta lukuisat ongelmat on tunnistettu ja projekti etenee nopeasti. Tämän jälkeen järjestelmään aletaan lisäämään lisäominaisuuksia sisältäviä moduuleita sen mukaan, mitä asiakkaat ovat eniten toivoneet.

6 POHDINTA

Mikäli voisin aloittaa projektin alusta, aloittaisin projektin tarkemmilla määrityksillä. Järjestelmän määrittely venyi, koska projekti aloitettiin, kun suurin osa työntekijöistä oli kesälomalla. Tämän takia oli vaikea saada usealta nykyistä järjestelmää toteuttaneelta henkilöltä mielipiteitä ja palautetta samaan aikaan tai ollenkaan. Tästä syystä tietyt määritykset jouduttiin määrittelemään uusiksi avainhenkilön palattua lomiltaan.

Projektissa käytettiin myös lukuisia uusia tekniikoita, joiden opiskelu ja ymmärtäminen vei aikaa varsinaiselta toteutukselta. Toisaalta opittuja tekniikoita voidaan hyödyntää tässä ja tulevissa projekteissa. Onneksi projektiin osallistuneilla Tamtron Solutions Oy:n työntekijöillä oli eriasteista tietoa suurimmasta osasta tekniikoista ja heiltä sai paljon oppia sekä apua ongelmatilanteihin.

Projektin aikana ilmeni, kuinka tärkeä on miettiä kehitettävien ohjelmien kansiorakenne kuntoon versiohallintaan. Kansiorakenteesta voi koitua paljon ylimääräistä vaivaa ohjelmoijalle, kun kehitettävän ohjelman käyttämiin moduuleihin tulee päivityksiä. Tämän projektin aikana vastaavia ongelmia ilmeni muutama, mutta kansiorakennetta pohdittiin uusiksi näiden tilanteiden minimoimiseksi.

Opinnäytetyöprojektia toteutettaessa tuli ilmi määrittelyn ja dokumentaation tärkeys ja se, miksi alan tunteminen on tärkeää järjestelmiä suunnitellessa. Asiakaslähtöisyyskin on huomioitu hyväksi ominaisuudeksi, sillä asiakkailta usein kuulee heille myydyn järjestelmän hyvät sekä huonot ominaisuudet, oli järjestelmä mikä hyvänsä. Näitä tietoja voidaan tuotekehityksessä hyödyntää rakentamalla järjestelmää, joka paikkaa muissa järjestelmissä ilmenneitä ongelmia.

Järjestelmän kehitysvaiheessa tuli paljon eri ominaisuuksia, jotka haluttiin järjestelmään. Ikävä kyllä moni näistä ominaisuuksista oli ristiriidassa jo määriteltyjen ominaisuuksien kanssa. Moni uusi ominaisuus katsottiin liian vaikeaksi toteuttaa ja ominaisuus päätettiin hylätä. Joissakin tapauksissa uusi ominaisuus katsottiin tärkeämmäksi kuin vanha ja ominaisuuksia määriteltiin uusiksi.

7 LÄHTEET

CHAPPELL, David 2005-02. Introducing Indigo: An Early Look [Viitattu 2013-12-18] Saatavissa http://msdn.microsoft.com/en-us/library/aa480188.aspx

ENTITYFRAMEWORKTUTORIAL.NET. What is Entity Framework? [Viitattu 2014-01-05] Saatavissa http://www.entityframeworktutorial.net/what-is-entityframework.aspx

GUTHRIE, Scott 2006-12-04. Announcing the release of the first "WPF/E" CTP [Viitattu 2014-01-12] Saatavissa http://weblogs.asp.net/scottgu/archive/2006/12/04/announcing-the-release-of-the-first-wpf-e-ctp.aspx

HANSELMAN, Scott 2010. 2010 Survey Results: What .NET Framework features do you use? [Viitattu 2013-12-16] Saatavissa http://www.hanselman.com/blog/2010SurveyResultsWhatNETFrameworkFeaturesDoYouUse.aspx

INFOWORLD 2012-07-20. Microsoft open-sources Entity Framework. [Viitattu 2014-01-6] Saatavissa http://www.infoworld.com/d/application-development/microsoft-open-sources-entity-framework-198213

JANSSEN, Cory. Windows Forms [Viitattu 2013-12-20] Saatavissa http://www.techopedia.com/definition/24300/windows-forms-net

JETBRAINS 2013. JetBrains Company History and Timeline [Viitattu 2014-01-08] Saatavissa http://www.jetbrains.com/company/history.jsp

JGAUFFIN 2013-02-11. How Resharper rocks my average work day [Viitattu 2014-01-07] Saatavissa http://www.codeproject.com/Articles/543162/HowplusResharperplusrocksplusmyplusaveragepluswork

MICROSOFT 2003-04. Programming Windows Forms with Managed Extensions for C++ [Viitattu 2014-3-10] Saatavissa http://msdn.microsoft.com/en-us/library/aa290064(v=vs.71).aspx

MICROSOFT 2009-04-11, Update on the Entity Framework in .NET 4 and Visual Studio 2010 [Viitattu 2014-03-10] Saatavissa http://blogs.msdn.com/b/adonet/archive/2009/05/11/update-on-the-entity-framework-in-net-4-and-visual-studio-2010.aspx

MICROSOFT 2012-08-02. Service Identity and Authentication. [Viitattu 2014-01-10] Saatavissa http://msdn.microsoft.com/en-us/library/ms733130(v=vs.110).aspx

MICROSOFT 2012-08-02. What is Microsoft Communication Foundation? [Viitattu 2013-12-20] Saatavissa: http://msdn.microsoft.com/en-us/library/ms731082(v=vs.110).aspx

MICROSOFT 2013. Overview of the .NET Framework. [Viitattu 2013-12-16] Saatavissa: http://msdn.microsoft.com/en-us/library/zw4w595w.aspx

MICROSOFT. Contrasting Silverlight and WPF [Viitattu 2014-01-12] Saatavissa http://msdn.microsoft.com/en-us/library/ff921107(v=pandp.20).aspx

MICROSOFT. Introduction to Windows Presentation Foundation [Viitattu 2014-03-10] Saatavissa http://msdn.microsoft.com/en-us/library/aa970268(v=vs.85).aspx

MICROSOFT. Learn more about .NET Framework support lifecycle [Viitattu 2014-03-10] Saatavissa http://support.microsoft.com/ph/548

MICROWAY. Visual Studio 2012 – Compare Editions [Viitattu 2013.12.23] Saatavissa http://www.microway.com.au/microsoft/visual-studio-2012-compare-editions.php

NOORUL, Hasan 2012-09-27. History of C# Programming [Viitattu 2013-12-17] Saatavissa http://aboutcsharpprogramming.blogspot.fi/2012/09/history-of-c-programming.html

S.SOMASEGAR 2012-08-01. Visual Studio 2012 and .NET 4.5 Complete! [Viitattu 2013-12-22] Saatavissa http://blogs.msdn.com/b/somasegar/archive/2012/08/01/visual-studio-2012-and-net-4-5-complete.aspx

THURROTT, Paul 2012-05-18. What I'm doing in Redmond [Viitattu 2013-12-22] Saatavissa http://windowsphonesecrets.com/2010/05/18/what-im-doing-in-redmond/

LIITE 1: OPINNÄYTETYÖSUUNNITELMA

Opinnäytesuunnitelma

Työajanseuranta järjestelmä

Juho Korhonen
ETX9SP

12/11/2013

Sisällysluettelo

VERSIOHISTORIA

VERSIO	PVM	MITÄ	KUKA
0.1	12.11.2013	• Ensimmäinen versio	Juho Korhonen
0.2	20.11.2013	• Lisätty versiohistoria kappale • Korjattu kirjoitusvirheitä • Liitetty aikataulutus	Juho Korhonen
0.3	25.11.2013	• Lisätty budjetointi kappale • Muutettu kaikki otsikot mustiksi • Lisätty 2. asteen otsikot sisällysluetteloon	Juho Korhonen
0.4	26.11.2013	• Poistettu otsikoiden edessä olevat merkit word:llä.	Juho Korhonen
0.5	1.12.2013	• Versiohistoria muutettu taulukkoon • Laitettu ensimmäisen tason otsikot omalle sivulleen • Korjattu tesktejä ja ulkoasua • Päivitetty "Projektin ohjaaminen" kappaleen tekstejä	Juho Korhonen
0.6	12.12.2013	• Taulukoiden otsikot siirretty näiden yläpuolelle • Muutettu versiointi tapaa • Korjattu kirjoitusvirtheitä	Juho Korhonen
1.0	19.1.2014	• Korjattu versiohistorian otsake	Juho Korhonen
1.1	6.3.2014	• Päivitetty päivämäärät	Juho Korhonen
2.0	14.3.2014	• Hyväksytty	Sami Lahti

1 JOHDANTO

1.1 Yleiskuvaus tästä dokumentista

Tässä dokumentissa löytyy yleistietoa tehtävästä opinnäytetyöstä. Tämän lisäksi tämä dokumentti sisältää opinnäytetyön aikataulutuksen ja budjetin. Lopputuotoksena valmistuu kirjoitelma työajan seurannasta ja sitä mittaavista järjestelmistä.

Opinnäytetyönä kehitettävä järjestelmä pyritään saamaan mahdollisimman valmiiksi opinnäytetyön aikana, mutta projektin laajuuden takia tätä ei voida taata. Tästä johtuen on mahdollista, että raportti kirjoitetaan vielä keskeneräisestä järjestelmästä

1.2 Tuote

Opinnäytetyönä toteutettavana järjestelmä on modulaarinen työajanseuranta järjestelmä, joka tallentaa ja lukee tietoa tietokannasta. Järjestelmän osia ovat muun muassa tietokanta, tietoliikennerajapinta, työpöytäsovellus sekä leimauspääte. Järjestelmää on tarkoitus jatkossa laajentaa sisältämään lisäosia esimerkiksi web-sovellus.

Lisäksi tuotetaan monisivuinen kirjoitelma, jossa tarkastellaan työajan hallintaa/valvontaa sekä laitteita/tekniikoita millä edellä mainittu toteutetaan. Tämän lisäksi kirjoitelmassa tullaan käymään läpi työajan hallintaan/valvontaan liittyviä osa-alueita, mikäli tiedolla on tärkeä rooli työajan hallinta/valvonta järjestelmien/tekniikoiden toteutukseen.

Opinnäytetyönä valmistettava työajanseuranta järjestelmä kehitetään .Net ympäristöön. Järjestelmä tullaan rakentamaan moduuleista, jolloin järjestelmään on helppo lisätä ja uusia osia.
Opinnäytetyössä käytettyjä tekniikoita ovat mm. C#, WCF sekä WPF.

1.3 Suunnitelman ylläpito

Suunnitelma ei ole lopullinen tuotos ja se tulee elämään opinnäytetyön rinnalla.

1.4 Määritelmät, termit ja lyhenteet

Alla olevassa taulukossa (TAULUKKO 1) selvennetään tässä dokumentissa ilmeneviä termejä.

TAULUKKO 1. Termistö

Termi	**Selitys**
Kirjoitelma	Opinnäytetyönä tehtävästä järjestelmästä kirjoitettava kirjoitelma

Työajan hallinta/valvonta Järjestelmä	Järjestelmä, jonka avulla yksittäisen tai useamman ihmisen työajan käyttöä voidaan valvoa, hallita ja/tai suunnitella. Ominaisuuksia muun muassa ovat: Työhön saapumisen ja työstä lähtemisen kirjaus, eri töihin käytetyn ajan kirjaus sekä työaika raportointi
WCF	Lyhenne sanoista Windows Communication Foundation. Yhteysrajapinta kirjasto joka on osa .NET Framework ohjelmistokehystä
WPF	Lyhenne sanoista Windows Presentation Foundation. Käyttöliittymä komponentti kirjasto joka on multimedia ominaisuuksiltaan laaja
C#	Microsoftin .NET Framework ohjelmistokehykseen kehittämä ohjelmointikieli

2 PROJEKTIN ORGANISOINTI

2.1 Projektin vaiheistus

Alla oleva taulukko (TAULUKKO 2) sisältää projektin vaiheet. Tarkempi aikataulutus käydään läpi projektin vaiheistus ja aikataulutus kappaleessa.

TAULUKKO 2. Aikataulutus lyhyesti

Vaihe	**Vaiheen nimi**
1	Suunnittelu
2	Suunnitelman esittely projektiin osallistuville
3	Suunnitelman puhtaaksi kirjoitus
4	Suunnitelman hyväksyminen
5	Aihealueiden rajaus
6	Opinnäytetyön rungon luonti
7	Rungon esittely
8	Tiedon keruu/kirjaus
9	Opinnäytetyön kirjoitus
10	Opinnäytetyön raa'an version esittely
11	Virheiden korjaus
12	Syyttömien rankaisu
13	Puhtaaksi kirjoitetun opinnäytetyön esittely
14	Loppu hiominen
15	Opinnäytetyön päättäminen

2.2 Vastuuhenkilöt

Alla olevassa taulukossa (TAULUKKO 3) on käyty läpi projektiin kuuluvat henkilöt.

TAULUKKO 3. Projektiin kuuluvat henkilöt

Nimi	**Puhelinnumero**	**Toimenkuva**
Juho Korhonen	050 571 2207	(Tekijä)
Sami Lahti	044 785 6337	(1. Ohjaaja)
Jussi Koistinen	044 785 5512	(2. Ohjaaja)
Joakim Heikkola	044 510 0049	(Työpaikan ohjaaja)

3 PROJEKTIN OHJAAMINEN

3.1 Tavoitteet ja priorisointi

Päätavoitteena on tuottaa opinnäytetyönä tehtävä järjestelmä Tamtron Solutions Oy:lle. Tavoitteena ei ole lopullinen versio lyhyen aikataulun takia, vaan järjestelmää rakennetaan niin pitkälle kuin mahdollista.

Toisena tavoitteina on tuottaa kirjoitelma, jossa käsitellään ja vertaillaan työajan seurantaan käytettyjä järjestelmiä sekä tekniikoita. kirjoitelma tullaan kirjoittamaan niin neutraalisti kuin saatavilla oleva tieto antaa periksi.

3.2 Oletukset, riippuvuudet, reunaehdot

Opinnäytetyötä kirjoitetaan pääasiassa Open Office ohjelmistolla, joten tiedon muotoilussa voi ilmetä ongelmia ennen viimeisintä versiota. Mikäli aika ei riitä jatkamaan opinnäytetyönä kehitettävää järjestelmää, tulee kirjoituksen kohteena olemaan jo valmistettu järjestelmän osuus.

3.3 Riskien hallinta

Tässä taulukossa (TAULUKKO 4) on käyty läpi mahdollisia riski tilanteita ja kuinka niihin on varauduttu.

TAULUKKO 4. Riskien hallinta

Riski	**Toimenpide**
Myöhästyminen	Pyritään kirimään aikataulu kiinni seuraavina päivinä.
Aikataulu ei pidä	Aikataulutusta tutkitaan uudelleen
Tietokone hajoaa	Työ on varmuuskopioitu ja sitä voidaan jatkaa toisella koneella
Oikeinkirjoitus jää pahasti jälkeen	Tarvittaessa oikeinkirjoituksen tarkastusväliä voidaan tihentää

3.4 Seuranta ja ohjaus

Opinnäytetyön tekijä sekä työpaikan vastaava ovat päivittäisessä kontaktissa opinnäytetyön kirjoitelman sekä opinnäytetyönä valmistettavan järjestelmän osalta. Tarkempi palaveri pidetään kahden viikon välein. Opinnäytetyön etenemisestä raportoidaan koulun ohjaajalle kirjallisella raportilla vähintään kahden viikon välein.

4 TEKNIIKKA

4.1 Menetelmät ja työkalut

Opinnäytetyönä valmistettava järjestelmä sekä kirjallinen osuus tullaan toteuttamaan täysin tietokoneellisesti. Varsinaisen opinnäytetyönä toteutettavan järjestelmän tekemiseen käytetään ohjelmointia helpottavia työkaluja ja järjestelmiä.

Kirjallinen osuus kirjoitetaan Open Office työkaluilla ja viimeistellään Microsoft tuoteperheen työkaluilla. Kirjallinen osuus kirjoitetaan aluksi yleisesti ja karkealla tasolla, jonka jälkeen aihealueita tullaan laajentamaan ja kirjoittamaan paljon syvällisemmin.

4.2 Dokumentointi

Opinnäytetyön etenemisestä kirjoitetaan raportteja, joissa käydään läpi tehdyt toimenpiteet, tulevat toimenpiteet ja tarkastellaan aikataulutusta ja sen pitävyyttä

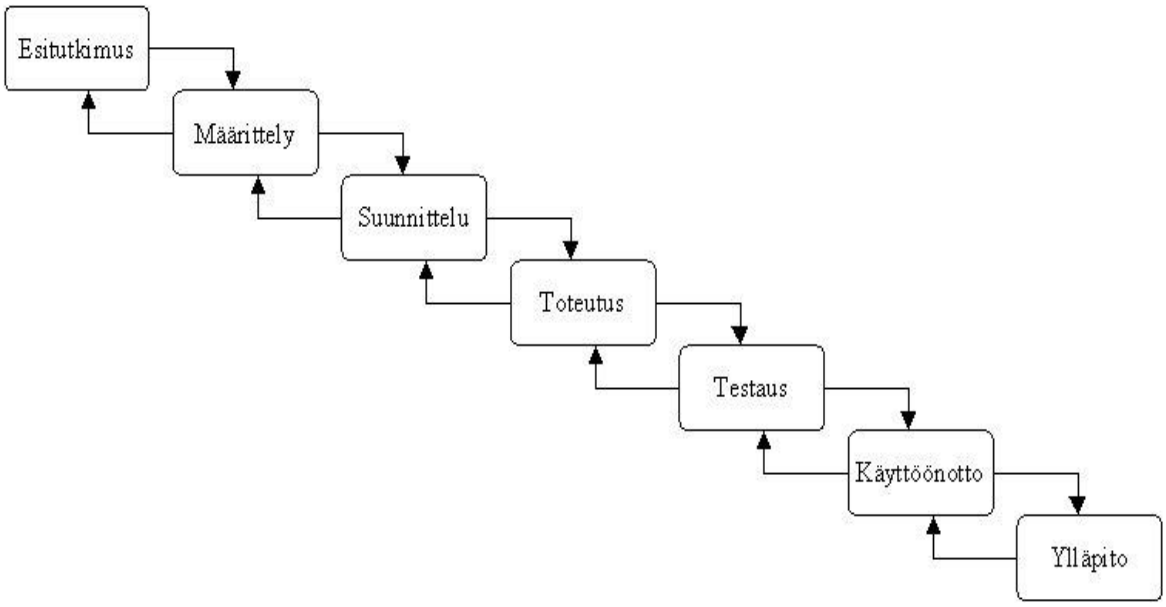
4.3 Laadunvarmistus

Opinnäytetyöhön käytetään vain luotettavia lähteitä ja lähteet merkitään selvästi. Tämän lisäksi lähteitä valittaessa pyritään mahdollisimman neutraaleihin lähteisiin. Opinnäytetyö kirjallisen osuuden oikeinkirjoitusta tarkastelee useampi henkilö ja tekstinkäsittely ohjelmistossa on käytössä oikeinkirjoitus.

5 PROJEKTIN VAIHEISTUS JA AIKATAULUTUS

5.1 Projektin osittaminen

Projektiin kuuluu varsinainen työtehtävä eli työajanseurantajärjestelmä sekä opinnäytetyön kirjallinen osuus. Projekti on ositettu vesiputousmallimaisesti (KUVIO 1). Aluksi tehdään yksi osuus, tämän jälkeen aloitetaan seuraava osuus. Mikäli tehdyssä osuudessa ilmenee puutteita ja/tai vikoja korjataan puutteet ja/tai viat ensin ja tämän jälkeen jatketaan seuraavaan osaa.



KUVIO 1. Vesiputousmalli

5.2 Projektin vaiheet

Alla olevassa taulukossa (TAULUKKO 5) on käyty läpi projektin vaiheita tarkasti päivämäärineen ja kuvauksineen

TAULUKKO 5. Vaiheistus ja aikataulutus

Vaiheen nimi	**Aloittamis päivämäärä**	**Kuvaus**	**Valmistumis päivämäärä**
Suunnittelu	10.11.2013	Suunnitelma kirjoitus ja laatiminen alkaa	10.11.2013
Suunnitelman esittely projektiin osallistuville	14.11.2013	Suunnitelman ensiverio esitellään koulun sekä työpaikan ohjaushenkilöille ja puutteet kirjataan ylös	14.11.2013

Suunnitelman puhtaaksi kirjoitus	21.11.2013	Suunnitelmassa havaittujen puutteiden korjaus ja uusi esittely	20.11.2013
Suunnitelman hyväksyminen	28.11.2013	Suunnitelman Lopullisen version lukkoonlyönti	19.1.2014
Aihealueiden rajaus	23.11.2013	Aletaan rajata opinnäyteyön aihealuetta. Kirjataan aihealueet ylös	23.11.2013
Opinnäytetyön rungon luonti	29.11.2013	Opinnäytetyön kirjoitettavan osan rungon valmistelu ja aihealueiden kirjoittaminen	25.11.2013
Rungon esittely	13.12.2013	Tehdyn runko esitellään ohjaushenkilöille	13.12.2013
Tiedon keruu/Kirjaus	20.12.2013	Kerätään tietolähteitä opinnäytetyön kirjallistaosutta varten	15.12.2013
Opinnäytetyön kirjoitus	6.1.2014	Opinnäytetyön kirjoittaminen alkaa. Aihealueet on rajattu ja runko hyväksytetty	16.12.2013
Opinnäytetyön raa'an version esittely	20.1.2014	Kirjoitusvirheiden esittely ohjaajille alkaa	13.1.2014
Virheiden korjaus	27.1.2014	Yllä mainittujen virheiden korjaus	14.1.2014
Puhtaaksi kirjoitetun	4.2.2014	Puhtaaksi kirjoitettu opinnäytetyö esitetään ohjaajille	19.1.2014

opinnäytet yön esittely			
Loppu hiominen	10.2.2014	Viimeiset viimeistelyt opinnäytetyöhön	28.2.2014
Opinnäyte työn esittely	22.2.2014	Esitellään opinnäytetyö seminaarissa	22.2.2014

5.3 Raportointi koulun ohjaajille

Opinnäytetyön etenemisestä kirjoitetaan kirjallinen raportti ohjaajille alla mainitun aikataulun (TAULUKKO 6) mukaisesti.

TAULUKKO 6. Raportoinnin aikataulutus

Raportti	**Raportointi päivämäärä**	**Valmistumis päivämäärä**
1. Raportti	25.11.2013	1.12.2013
2. Raportti	9.12.2013	13.12.2013
3. Raportti	20.12.2013	20.12.2013
4. Raportti	6.1.2014	13.1.2014
5. Raportti	20.1.2014	-
6. Raportti	3.2.2014	-
7. Raportti	17.2.2014	-

5.4 Äidinkielen tarkastus

Opinnäytetyö lähetetään äidinkielen opettajalle tarkastettavaksi alla mainitun aikataulun (TAULUKKO 7) mukaisesti.

TAULUKKO 7. Äidinkielen tarkastus

Tarkastus	**Tarkastus päivämäärä**	**Valmistumis päivämäärä**
1. Tarkastus	20.1.2014	25.1.2014
2. Tarkastus	8.2.2014	6.3.2014

5.5 Englannin tarkastus

Opinnäytetyön englannin kielen tarkastus tapahtuu alla mainitun aikataulun (TAULUKKO 8) mukaisesti.

TAULUKKO 8. Englannin tarkastus

Tarkastus	**Tarkastus päivämäärä**	**Valmistumis päivämäärä**
1. Tarkastus	8.2.2014	17.2.2014

6 BUDJETTI

6.1 Henkilöstöresurssit

Taulukossa (TAULUKKO 9) on käyty läpi projektiin kuuluvat henkilöresurssit. Tämän lisäksi projektilla on yksi varahenkilö, mikäli aikataulu jää pahasti jälkeen tai tulee pitkäaikaisia sairastumisia.

TAULUKKO 9. Henkilöresurssit

Henkilö	**Tiedot**
Juho Korhonen	Tuleva insinööri ei paljoa aiempaa työkokemusta. Vastaa käyttöliittymän suunnittelusta ja toteuttamisesta. Opinnäytetyön tekijä
Lauri Kare	Hän on ollut alalla jo muutaman vuoden. Insinööri. Vastaa tekniikoiden toteuttamisesta ja serveri puolesta

6.2 Arvioitu jäljellä oleva työmäärä

Alla olevissa taulukoissa (TAULUKKO 10 ja TAULUKKO 11) on käyty läpi projektia työstävien henkilöiden jäljellä olevat työmäärät. Työtunnit ovat arvioita ja niihin on sisällytetty henkilön tehokkuus sekä työn alustavaan tutkimiseen ja selvitykseen kuluva aika.

TAULUKKO 10. Työmääräarvio, Lauri Kare

Työtehtävä	**Tarvittava aika tunteina**	**Tekijä**
WCF	15	Lauri Kare
Tietokanta	15	Lauri Kare
Tiedonsiirto	7,5	Lauri Kare
Laskenta	38	Lauri Kare
Lisenssöinti	7,5	Lauri Kare
Ohjelmistopäivitys	38	Lauri Kare
Testaus	76	Lauri Kare
Korjaus	76	Lauri Kare
Yhteensä	**273**	

TAULUKKO 11. Työmääräarvio, Juho Korhonen

Työtehtävä	**Tarvittava aika tunteina**	**Tekijä**
Käyttöliittymä moduulit		
Käyttäjät	7,5	Juho Korhonen
Tapahtumat	7,5	Juho Korhonen
Työpäivät	15	Juho Korhonen
Työpäivien hyväksyntä	38	Juho Korhonen
Yritykset/Osastot	38	Juho Korhonen
Testaus	76	Juho Korhonen
Korjaus	76	Juho Korhonen
Yhteensä	**258**	
Opinnäytetyö		
Kirjoitus	130	Juho Korhonen
Korjaus	50	Juho Korhonen
Yhteensä	**180**	

6.3 Muut resurssit

Käytössä on Visual Studio 2012 ohjelmisto versio sekä ReSharper ohjelmisto avustamassa sekä koodin puhtauden ylläpitoa että sen kirjoittamisnopeutta.

Testiympäristöä varten käytössä on testikone ja testikäyttöön tarkoitettuja lisenssejä, joilla voidaan simuloida eriasteisia käyttöympäristöjä. Tämän lisäksi testi lisensseillä voidaan tutkia kuinka eri ohjelmat kuten Microsoft SQL Server 2008/2012 käyttäytyvät työajanseurantajärjestelmän osana.