LAUREA AMMATTIKORKEAKOULU Uuden edellä

Windows Installer XML: käyttöönotto asennusohjelman tekemisessä

Koskina, Ekaterina

2012 Kerava

Laurea-ammattikorkeakoulu
Kerava

Windows Installer XML: käyttöönotto asennusohjelman tekemisessä

Ekaterina Koskina
Tietojenkäsittelyn koulutusohjelma
Opinnäytetyö
Maaliskuu, 2012

Sisällys

1 Johdanto 6
2 Asennusohjelmat Windows- käyttöjärjestelmissä 6
2.1 Windows Installer 7
2.1.1 Synty, kehitys ja nykytilanne 7
2.1.2 MSI- asennustiedoston rakenne 8
2.2 WiX- työkalupaketti ja sen kilpailijat 9
3 WiX:in käyttöönotto 11
4 WiX- teknologia 13
4.1 Rungon Elementit 14
4.1.1 Tuote- elementti 15
4.1.2 Pakkaus- elementti 15
4.1.3 Media- elementti 16
4.2 Hakemistot 17
4.3 Komponentti- elementit 17
4.4 Vaihtoehto- elementit 18
4.5 Käyttöliittymät 20
5 Käytännön esimerkki asennusohjelmasta 21
5.1 Lähtökohdat 21
5.2 Asennusohjelman teko 22
5.2.1 Perustiedot koodiin 22
5.2.2 Asennettavat tiedostot 23
5.2.3 Poistaminen 25
5.2.4 Hyperlinkkikuvake 26
5.2.5 Asennuksen vaihtoehdot 27
5.2.6 Käyttöliittymän luominen 28
5.3 Valmiin asennusohjelman kokoonpano ja testaus 29
5.3.1 Orca 30
5.3.2 Asennus 31
5.4 Arvio 35
6 Yhteenveto 35
Lähteet 36
Liitteet 38

Laurea-ammattikorkeakoulu
Kerava
Tietojenkäsittelyn Koulutusohjelma

Tiivistelmä

Ekaterina Koskina

Windows Installer XML: käyttöönotto asennusohjelman tekemisessä

Vuosi 2012 Sivumäärä 35

Opinnäytetyössä opittiin luomaan oma asennusohjelma Windows- käyttöjärjestelmää varten käyttäen Windows Installer XML työkalupakettia. Tavoitteena oli tutustuttaa lukijoita Window Installer- palveluun, joka vastaa ohjelmasovelluksien asennuksesta ja paikannuksesta Windows- käyttöjärjestelmissä, ja sen kautta Windows Installer XML- teknologiaan, jota käytetään asennuksien toteuttamiseksi. Teknologian tutustumisen perusteella jokainen lukija oppii luomaan oman haluamansa asennusohjelman.

Opinnäytetyössä edettiin kertomalla ensiksi Windows- käyttöjärjestelmän asennusohjelmien tekoon liittyvistä käytännöstä, Windows Installer palvelusta ja MSI- asennuspakettitiedostoista. Ilman näitä tietoja olisi vaikea ymmärtää Windows Installer XML teknologian avulla luotujen asennuspakettien rakennetta ja tarkoitusta. Mainittiin sekä verrattiin keskenään Windows Installer XML ja muut tavat luoda asennusohjelmaa Windows- käyttöjärjestelmää varten. Kerrottiin mahdollisista työkaluista kuten Microsoft Visual Studio ja SharpDevelop sekä niiden saantimahdollisuuksista. Windows Installer XML:n teorian kertomisen jälkeen näytettiin miten läpikäydyillä työkaluilla toteutettiin asennuspaketin luomisen esimerkki käyttäen siitä opittuja perustietoja teknologiasta. Esimerkissä pystyttiin luomaan asennuspaketti, joka ei toiminnoiltaan eikä ulkonäöltään erottuisi markkinoilla jaettavista tuotteista.

Asennusohjelman luomisen esimerkillä Windows Installer XML – teknologia yritettiin havainnollistaa lukijoille. Opinnäytetyön tuloksena saatiin asianmukainen tietolähde kaikille teemasta kiinnostuneille.

Asiasanat: Windows Installer XML, WiX, MSI,Windows Installer, asennusohjelma

Laurea University of Applied Sciences
Kerava
Business Information Technology

Abstract

Ekaterina Koskina

Windows Installer XML: Implementation in the Process of Creating an Installation Program

Year 2012 Pages 35

This Bachelor's thesis describes how to create an installation program for Windows operating system using Windows Installer XML toolset. The goal was to familiarize readers with Windows Installer service, which is responsible for program installation and positioning applications in Windows operating systems, and through it with Windows Installer XML technology, which is used to implement installations. On the basis of this knowledge readers will learn how to create their own preferred installation program.

First, this study describes details of creating installation programs in the Windows operating system environment, Windows Installer and MSI installation packages. Without this information it would be difficult to understand the structure and purpose of installation package files that were created using Windows Installer XML technology. Other ways of creating installation programs for Windows operating system are also mentioned and compared to Windows Installer XML toolset. Possible tools such as Microsoft Visual Studio and SharpDevelop are described with references how to get them. After Windows Installer XML theory, it is shown how to use learned technologies basics and tools in an example of installation package creation. The outcome installation program does not functionally and visually differ from products distributed on market.

The purpose of creation of the example was to help readers learn Windows Installer XML technology by visualizing it for them. As a result of this thesis everyone who is interested in this theme can get a due information source.

Keywords: Windows Installer XML, WiX, Windows Installer, MSI, installation program

1 Johdanto

Jokainen sovellus, kaupallinen tai ei, tarvitsee nykyään oman asennusohjelman asennettavaksi loppukäyttäjän tietokoneelle. Harva loppukäyttäjä miettii miten sovelluksen asennusprosessi hänen tietokoneella tapahtuu ja miten sovelluskehittäjät sen toteuttavat. Tämän opinnäytetyön tehtävä on tutustuttaa ihmisiä Windows Installer XML- (alla tekstissä lyhennettynä WiX) teknologiaan, mitä käytetään asennuksien toteuttamiseksi. Se tapahtuu ensiksi kertomalla Window Installer- palvelusta, mikä vastaa ohjelmasovelluksien asennuksesta ja paikannuksesta Windows- käyttöjärjestelmissä, jonka jälkeen vasta siirrytään WIX- teknologian oppimiseen. Teknologian perusteet ovat helpot sekä mahdolliset käytetyt työkalut ongelmitta saattavissa. Mikään ei estä jokaista kokeilemasta luoda omaa asennusohjelmaa jaettaville tiedostoille.

Opinnäytetyön rakenne on toimintakeskeinen. Käydään ensiksi WiX- teoria läpi ja sitten kuvataan asennusohjelman suunnittelu ja kehitys keksittyä yrityksen sovellusta varten. Tällä esimerkillä WiX – teknologia yritetään havainnollistaa lukijalle.

Kiinnostus juuri tämän aiheen valintaan opinnäytetyöksi kasvoi tutkittaessa kaikkea siihen liittyvää aineistoa. Ennen tätä työtä teeman mukaisia julkaisuja ei ollut saattavilla suomenkielisinä. Joten opinnäytetyön tekijän tavoitteena on luoda selkeä, auttavainen lähde teemasta kiinnostuneille, joilla sitä ennen ei ollut mahdollisuuksia tutustua syvemmin aiheeseen aineistokielen vieraskielisyyden takia.

2 Asennusohjelmat Windows- käyttöjärjestelmissä

Vaikka asennusohjelmat ovat tulleet tutuiksi vuosien varrella tietokoneteknologian kehityksen myötä, Windows Installer:in tunteminen tai MSI- asennuspaketin tekeminen on ollut haastavaa. Asennusohjelmat ovat kuin pieniä tietokantoja, jotka sisältävät tiedon, miten tiettyjä sovelluksia tulee purkaa ja sijoittaa loppukäyttäjän tietokoneeseen. Tarve oman asennusohjelman luomisesta tulee esimerkiksi omien sovellusten kehitykseen syventyvillä yrittäjillä ja yrityksillä. Windows- perheen käyttöjärjestelmät ovat yleisimmät loppukäyttäjillä, joten ohjelman tulisi toimia ja osata asentaa ainakin Windows- järjestelmiin, jotta suurin mahdollinen käyttäjäryhmä ei jäisi väliin. Windows- järjestelmässä on sisäänrakennettu Windows Installer-komponentti, joka huolehtii ohjelmien asennusprosessista.

Sovelluksien kehittäjillä on muutama tapa lähteä liikkeelle. Ensimmäinen tapa on itse luoda asennusohjelma, joka muodostuu pienen tietokannan asennuspoluista ja tiedoista. Siihen tarvitaan hyvää Windows Installer API tuntemusta. Toinen tapa on ostaa asennusohjelmien tekoon erikoistunut ohjelma, kuten InstallShield, joka tekee lähes kaiken käyttäjän puolesta. Nämä ohjelmat huolehtivat pienistäkin yksityiskohdista, mutta eivät näytä käyttäjälle kaikkia

asennusohjelman luomisen vaiheita. Kolmas mahdollinen tapa esitellään tässä työssä. Windows Installer XML on kuin ohut kalvo ohjelmoijan ja Windows Installer- asennusohjelman välillä. Se ei laajenna mitenkään Windows Installerin toiminnallisuutta, mutta ei toisaalta piilotakaan mitään sen mahdollisuuksista. WiX:in tehtävä on helpottaa luomisvaihetta tekemällä sen ohjatuksi ja itsestään selväksi. WiX on tapa rakentaa asennusohjelma ja sitä on kohtuullisen helppo oppia hallitsemaan.

2.1 Windows Installer

Microsoft Windows Installer (MSI) on Windowsin asennus- ja konfigurointipalvelu. Se on ollut osa Windows- perheen järjestelmiä Windows 2000- ja Windows Me- järjestelmistä alkaen. Se on mahdollista asentaa myös Windows 95, Windows 98 ja Windows NT 4.0 järjestelmiin. Suurin tavoite teknologiaa luotaessa oli pienentää käyttökustannukset käyttäjille eli TCO (Total Cost of Ownership) mahdollistamalla ohjelmien ja sovelluksien tehokkaimman asennuksen ja konfiguroinnin.

2.1.1 Synty, kehitys ja nykytilanne

Windows Installer palvelun teknologia sai alkunsa 1990 - luvulla, Microsoftin DOS- käyttöjärjestelmän (Disk Operating System) aikana. Tuohon aikaan järjestelmät pystyivät tekemään vain yhtä tehtävää kerrallaan, eikä asennusta varten ollut mitään yhteistä käyttöliittymää. Oli yrityksiä luoda käyttöliittymä, mutta ne jäivät toiminnoiltaan keskeneräiseksi. Keskivertokäyttöliittymässä oli värikäs tausta ja siinä ikkunoita, joissa kysyttiin käyttäjältä tietoja. Ikkunat myös näyttivät asennusprosessin vaiheet. Kun Windows 3.00- järjestelmä tuli markkinoille, alkoi oikea asennusohjelmien käyttöliittymien kehitys. Silloin tietokoneen käyttöliittymien valmistajat eivät vielä tarjonneet integroituja ratkaisuja asennukseen. Tämä mahdollisti asennusohjelmien palveluita tarjoavien yritysten ilmestymisen markkinoille. Tunnetuimmat niistä ovat InstallShield Software ja WiseSolutions. Nämä yritykset toivat yhtenäisen käyttöliittymän asennusohjelmiin ja niillä oli suuri rooli loppukäyttäjän näkemyksen muodostumiseen siitä miltä asennusohjelman tulisi näyttää.

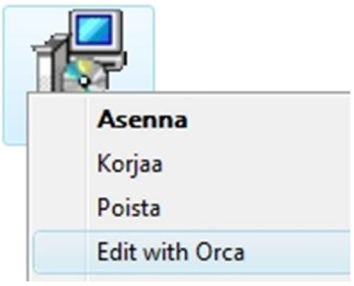
Suurimman läpimurron ohjelmien asennuksen hallinnoinnissa toi Windows 2000- käyttöjärjestelmä. Sen mukana tuli paljon uusia palveluja kuten esimerkiksi asennusten hallinnointiin liittyvät palvelut Windows File Protection ja Windows Installer. Molemmat teknologiat ovat tiiviisti yhdistetty. Windows File Protectionin tehtäviin kuuluu ohjelmien poistamisen tai muokkaamisen kieltäminen, jos ne ovat tärkeitä Windows- käyttöjärjestelmän toimimisen kannalta. Windows Installer huolehtii ohjelmiin kuuluvista prosesseista, kuten oikeasta asennuksesta, päivityksestä, korjaamisesta ja poistosta niin, että ne eivät vaikuta muiden ohjelmien toimivuuteen. Viimeisimmät versiot Windows Intaller:ista ovat tällä hetkellä 5.0 ja 4.5. Versio

5.0 toimii Windows 7 ja Windows Server 2008 R2 järjestelmissä. Vanhemmissa Windows XP, Vista Server 2003 ja Server 2008 järjestelmissä toimii versio 4.5.

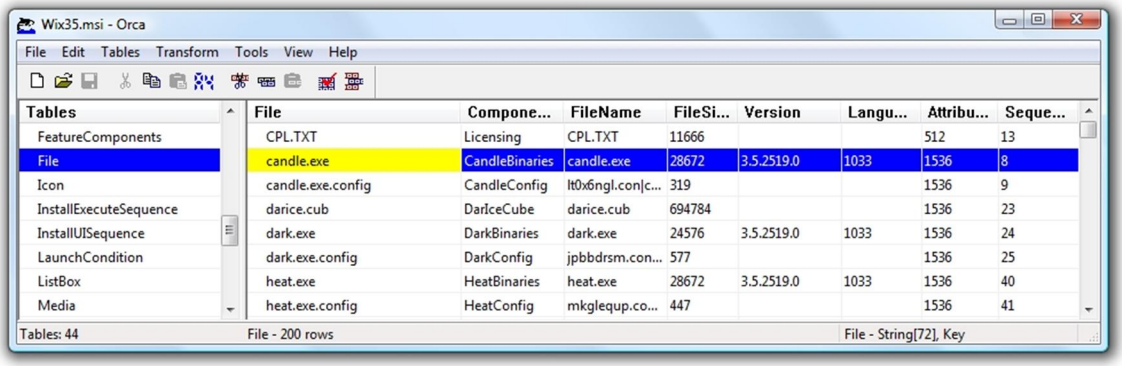
2.1.2 MSI- asennustiedoston rakenne

Yleensä asennuspaketit varastoidaan .msi- päätteisiin tiedostoihin, jotka ovat relaatiotietokantoja. Ne säilyttävät kaikki tarvittavat tiedot asennusta varten. Windows Installer hallinnoi pääsyä näihin tietoihin. Asennuspaketti koostuu monista toisiin liittyvistä taulukoista. Koska kyseessä on relaatiotietokanta, taulukot yhdistyvät perus- ja viiteavaimien avulla. Tämä takaa tehokkaan asennusprosessin valvomisen ja antaa käyttäjälle mahdollisuuden räätälöidä vaikeitakin ohjelmia tai joukon sovelluksia omien tarpeittensa mukaan. Taulukoihin kuuluvia tietoja ovat saattavilla olevat sovellusvaihtoehdot, lisäkomponentit, valittujen sovellusten ja lisäkomponenttien yhteydet, tarvittavat Windows- rekisterimerkinnät sekä asennusohjelman käyttöjärjestelmä.

Asennusohjelman luomista varten tulee taulukoihin syöttää tarvittavat tiedot tuotteesta ja sen asennuksesta. Tämän voi tehdä käsin eli käyttäen ORCA- työkalua, joka kuuluu Windows Installer SDK- työkalupakettiin. Ohjelma ei ole saattavilla erikseen, joten ensimmäiseksi pitää asentaa koko työkalupaketti koneeseen. Paketti on vapaasti ladattavissa Microsoftin sivuilta. Orca- sovelluksen pitää asentaa vielä erikseen. Oletuksena se asentuu polulle C:\Program Files\ Misrosoft SDKs\Windows\<Version numero>\Bin\Orca.exe. Taulukoiden täyttäminen käsin on hyvin työlästä jopa pienen asennusohjelman kohdalla. Ohjelma voisi olla hyödyllinen, kun tutkitaan asennuspaketin tietokantarakennetta. Orca:n saa avattua painamalla oikeata hiiren nappia jonkun asennussovelluksen päällä ja valitsemalla menuvalikosta "Edit with Orca" tai hakemalla Orca- ohjelmaa komentoriviltä. Esimerkkinä voi ottaa WiX- työkalupaketin sisältävän Wix35.msi- tiedoston (Kuva 1). Kun tiedosto avataan, tulee näkyville tämän asennuspaketin koostumus (Kuva 2).



Kuva 1. MSI-tiedoston avaus Orca- ohjelmalla.



Kuva 2. Wix35.msi- asennuspaketin sisältölista Orca- ohjelmassa.

Usein MSI -tiedostot eivät sisällä kaikkia toimintoja, joita loppukäyttäjä saattaisi tarvita. MSI-tiedostot ovat kuitenkin keskeisiä, koska muut toiminnot tapahtuvat niiden ympärillä. Lisäksi on olemassa MSM-, MSP- ja MST- tiedostot. MSM- tiedostot ovat liityntämoduulitiedostoja. MSM- tiedostot sisältävät joitain yksittäisiä tietoja, esimerkiksi kuvauksen yhdestä komponenttikokonaisuudesta, joka voidaan yhdistää MSI- asennuspakettiin osaksi tekovaiheessa. MST- tiedostot ovat muunnostiedostoja, jotka hallinnoivat asennuksen prosessin ja muuttavat MSI asennuspaketin arvoja. Käyttäen MST- tiedostoa, arvojen muutokset eivät tallennu itse MSI–tiedostoon, ne vain määräävät miten tietty asennus tapahtuu. MSP- tiedostot ovat päivitystiedostoja, jotka muuttavat MSI- tiedoston arvoja.

2.2 WiX- työkalupaketti ja sen kilpailijat

Asennusohjelmien tekoon on useita erikoistuneita ohjelmia. Asennus voi tapahtua siihen tarkoitetulla pakettihallinnon järjestelmällä (Package Management System), joka on sisäänrakennettu järjestelmään (kuten Windows Installer Windows- järjestelmissä tai RPM- ja APT Linux- käyttöjärjestelmässä) tai ohjelmiston mukana tulleen oman scriptin käyttämän asennusohjelman avulla. Asennusohjelmaa valittaessa käyttöjärjestelmien rakenteen erilaisuus on

otettava huomioon. On selvitettävä miten ja missä kehittämisympäristössä asennusohjelman teko onnistuu sekä missä käyttöjärjestelmissä loppukäyttäjät tulevat tarjottavia ohjelmia käyttämään. Alla esitellään lyhyet kuvaukset ja vertaileva taulukko WiX työkaluista, sekä yleisimmistä asennusohjelmien tekoon erikoistuneista ohjelmista, jotka toimivat Windows-perheen käyttöjärjestelmissä (Taulukko 1). Mainitut asennusohjelmat käyttävät erilaisia teknologioita. Jotkut käyttävät Windows Installer pakettihallinnon järjestelmää, toiset omia ratkaisujaan.

	Installshield	WIX	NSIS	Inno Setup
Hinta	€ 1789.00	Ilmainen	Ilmainen	Ilmainen
Lisenssityypi	Proprietary - Shareware / Patentoidu, Maksullinen, 21 päivän kokeiluversio saattavilla	CPL (Common Public License) Vapaa ohjelmisto / Avoin lähdekoodi	zlib / libpng License Jaetaan niin kun tuote on, muokattua koodia ei saa nimittää alkuperäiseksi	Inno Setup License / Free software / Vapaa ohjelmisto
Koko	215.8 MB	25.1MB	1.5MB	1.7MB
Teknologia	Oma / Windows Installer	Windows Installer	Oma	Oma

Taulukko 1. Windows- käyttöjärjestelmän asennusohjelmien valmistukseen erikoistuneiden ohjelmien kärkinelikon kilpailija-analyysitaulukko.

Windows Installer XML (WiX) on avoimen lähdekoodin työkalupaketti ja teknologia, jonka avulla luodaan asennusohjelmia Windows- tietokonejärjestelmässä toimintaan tarkoitettuihin sovelluksiin. WiX:in keksi sen johtava kehittäjä Rob Mensching. WiX:in laaja käyttäjäpiiri johtuu sen yksinkertaisuudesta. WiX antaa mahdollisuuden tutustua jokaiseen asennusohjelman luontivaiheeseen ja muokata sitä. Koodia voi kirjoittaa Muistiossa, käyttäen WiX:in tarjoamia työkaluja toimintojen lisäämiseksi tai muokaten valmista pohjaa tuetuissa sovelluskehitysalustoissa. Koodi on olemassa olevan XML kielen tapaista, joten se on helposti luettavissa kaikille, joille XML kieli on vähänkään tuttu. WiX:in etuna on se, että ei tarvitse hankkia kallista ohjelmistoa.

Vuonna 1992 kehitetty ja jatkuvassa kehityksessä oleva InstallShield on asennusohjelmien markkinoiden jättiläinen. Sen viimeisin versio ohjelmasta on InstallShield 2012. Ohjelmassa on käytössä oma scriptikieli InstallScript. InstallShield tukee myös Windows Installer- pakettihallinnon teknologiaa. InstallShield on kaikista muista asennusohjelman tekoon erikoistuneista ohjelmistoista suurin ja vaikein. Siitä löytyy suuri määrä asetuksia, jotka eivät ole edes tar-

peellisia pienen asennusohjelman luontia varten. Usein InstallShield otetaan käyttöön jotain isoa, monimutkaista jakelussa olevaa ohjelmistoa varten.

Nullsoft Scriptable Install System (NSIS) on avoimen lähdekoodin järjestelmä asennusohjelmien luomiseen. Alun perin NSIS kehitettiin Winamp- ohjelman asennusta varten. Kun tuote julkaistiin SourceForge.net sivustolla, tuotteen saivat käsiinsä myös ulkopuoliset ohjelmistokehittäjät. Monet tietokoneen peruskäyttäjälle tutut ohjelmat kuten: 7-zip, Mozilla Firefox, eMule, Miranda IM, VLC Player, Winamp jne, käyttävät asennusohjelmana NSIS:a.

Vuonna 1997 markkinoille tullut Inno Setup on edelleen laajasti käytetty. Inno Setupin kehittäjä on Jordan Russel. Inno Setup 5.3.0. uudistuksen jälkeen käytettävissä on kaksi versiota: Non Unicode Inno Setup ja Unicode Inno Setup.

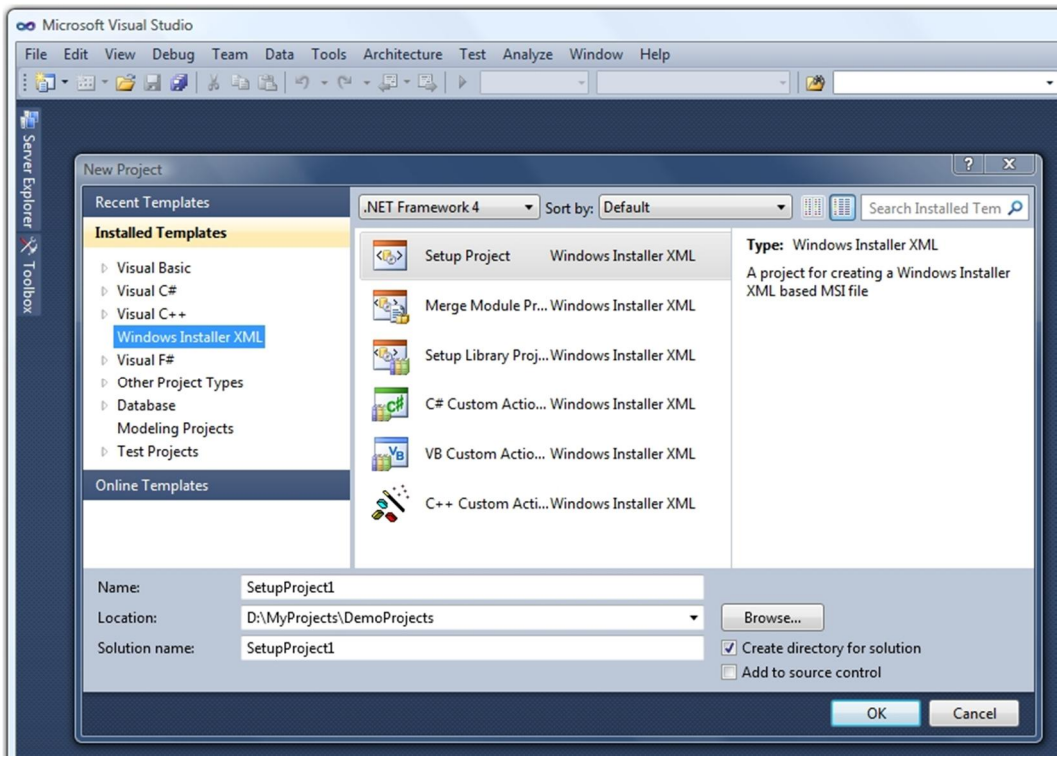
3 WiX:in käyttöönotto

WiX on avoimen lähdekoodin tuote, joten se on vapaasti jaettavissa Internet- verkossa. WiX:in käyttöönotto riippuu siitä, missä ympäristössä sitä tullaan käyttämään. Alla on ohjeet miten WiXin voi ottaa käyttöön kahdessa eniten WiX- asennusohjelmien tekoon erikoistuneessa kehitysympäristöissä: Visual Studiossa ja SharpDevelopissa. Molemmissa sovelluskehitysalustoissa voidaan toisessa tehdyt projektit avata ongelmitta. Kaikista ajantasaisimman version WiX-tuotteesta saa projektin virallisilta sivuilta: http://wix.sourceforge.net. Viimeisin valmis versio on tällä hetkillä 3.5.

Microsoft Visual Studio on sovelluskehitysalustojen tuotesarja, joka tukee useita ohjelmointikieliä. Viimeisin julkaistu versio on vuodelta 2010 ja sitä käytetään myöhemmin tässä tutkielmassa asennusohjelman tekemisen esimerkissä. Ohjelmisto on kaupallinen, mutta se on saattavilla kokeiluversiona Microsoftin sivuilla osoitteena: http://www.microsoft.com/download/en/details.aspx?id=12187. Jokaisessa MS Visual Studio paketissa on sisäänrakennettu ”Setup Project” perustoimintojen sapluuna, joka on myös tarkoitettu asennusohjelman tekemiseen. Setup Projectin avulla voi luoda asennusohjelman parilla hiiren painaluksella. Lisäksi se on integroitu MSBuild alustan kanssa. Visual Studion oma Setup Project ei ole yhtä monipuolinen kuin WiX. Siinä on rajoituksia asennuskäyttöliittymän laajentamisessa, vaikeuksia päivityksien ja lisäyksien tekemisessä eikä siinä ole mahdollista asentaa vain valittuja elementtiä asennuspaketista.

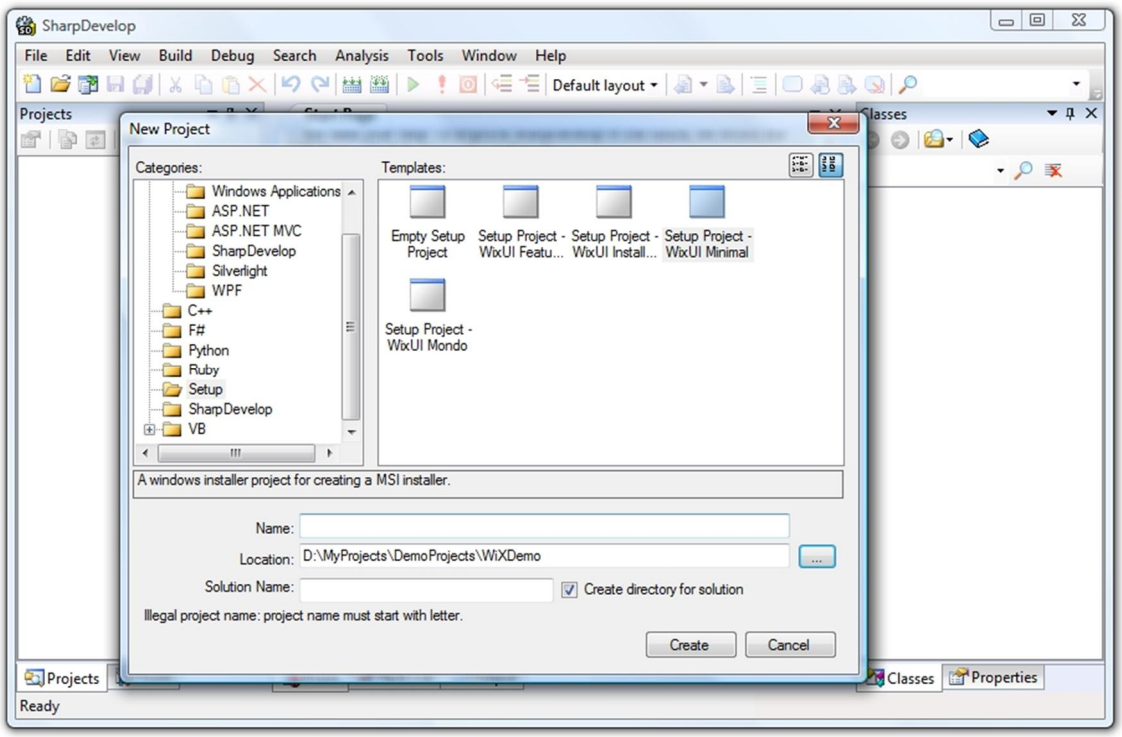
WiX työkalupaketin kanssa tulee Votive- apuohjelma, mikä toimii Visual Studion lisänä ja helpottaa WiX:in integraatiota Visual Studioon. Se lisää Visual Studioon syntaksikorostuksen, WiX IntelliSense toiminnon (Microsoft:in automaattinen täydennystoiminto), sekä projektien sapluunat. Votive ei toimi Visual Studion Express versiossa. Vuoden 2005 Standardin vanhemmissa

versioissa Votive'n toimimiseen tarvitaan ProjectAggregator2.msi lisäkomponentin asennusta. Sen saa ladattua WiX projektin sivulta: http://wix.sourceforge.net/redist/ProjectAggregator2.msi. Jos Wix35.msi paketti on asennettu oikein, Visual Studiota löytyy uusi vaihtoehto sapluuna: Windows Installer XML (Kuva 3).



Kuva 3. Windows Installer XML sapluunan ilmestyminen Visual Studion WiX- työkalupaketin asentamisen jälkeen.

SharpDevelop on avoimen lähdekoodin .Net- kielen sovelluskehitysalusta. Se tukee C#, Visual Basic .NET, Boo, IronPython, IronRuby, Clarion, F# ja C++ ohjelmointikieliä. Itse sovelluskehitysalusta on toteutettu C#- kielellä. Ensimmäinen beta- versio sovelluksesta ilmestyi vuonna 2000. Viimeisimmät saattavilla olevat versiot tällä hetkellä ovat 4.1 ja 4.2 beta versio. Se on vapaasti ladattavissa avoimen lähdekoodin ohjelmien jakelusivustolta: http://sourceforge.net/projects/sharpdevelop/. SharpDevelop toimimiseksi tarvitaan .NET Framework:in versio 2.0 tai uudempi. SharpDevelop tukee WiX- työkalupakettia ja on vaihtoehto ihmisillä, jotka eivät halua tai voi käyttää MS Visual Studiota. SharpDevelop muistuttaa Visual Studiota, mutta sen toiminnot ovat suppeammat (Kuva 4).



Kuva 4. SharpDevelop - sovelluskehitysalustan ulkonäkö ja sen WiX sapluunat.

4 WiX- teknologia

Wix on Windows XML-koodia käyttävä työkalupaketti asennuspakettien asentamiseen. Asennuspakettia käytetään jaettavan ohjelman (distribuution) asentamista varten. Itse jaettava ohjelma voi koostua erilaisista objekteista, kuten ajettavista tiedostoista, kirjastoista, dokumentaatioista jne. Kaikki nämä tiedostot kasataan komponentteihin, joita Windows Installer käyttää tiedostojen valvomiseen. Komponenteista voi rakentaa erilaisia kokonaisuuksia, jotka yhdistävät toisiinsa liittyvät toiminnot. Jaettavan ohjelman toimimisen kannalta pakolliset tiedostot, vapaaehtoiset lisäykset ja ohjetiedostot voidaan jakaa erilaisiin kokonaisuuksiin. Uutta teknologiassa on, että kaikki jaettavan ohjelman kuvaukset näkyvät kuten tavallinen XML-tiedosto, mihin on helppoa tehdä muutoksia.

WiX- työkalupaketissa on 12 työkalua: candle, light, lit, dark, heat, insignia, melt, torch, smoke, pyro, wixcop ja wixunit. (Taulukko 2) Suurin osa niiden nimistä on tuli-teeman mukaisia. Työkalupaketin nimikin WiX kuulostaa englannin kielellä sanalta wick, mikä on sytytyslanka. Kaikki työkalut ovat komentorivin kautta ajettavia. Asennusohjelman kirjoittamisen lähtökohtana on yksi WiX XML- tiedosto, minkä rakennetta kutsutaan skeemaksi. Yleensä WiX- tiedostot ovat .wxs- päätteisiä. Niiden ympärille rakentuvat kaikki toiminnot: muuntaminen väliobjekti tiedostoksi ja sen jälkeinen yhdistäminen light- työkalun avulla valmiiksi MSI / MSM-

tiedostoiksi tai toisinpäin käyttäen dark työkalua MSI / MSM- tiedostosta muuttaminen takaisin WiX- tiedostoksi.

Candle.exe	Muuttaa / Kääntää WiX- lähdetiedoston (.wxs) väliobjekti tiedostoksi (.wixobj).
Light.exe	Yhdistää wixobj- tiedostot luoden valmiit MSI- tai MSM- tiedostot.
Lit.exe	Yhdistää useita wixobj- tiedostoja WIX- kirjastoihin (.wixlib).
Darc.exe	Muuttaa / Kääntää MSI, MSM tiedostot WiX lähdetiedostoihin (.wxs).
Heat.exe	Luo WiX- lähdetiedoston (.wxs), mikä määrittää komponentit eri paikoista.
Insignia.exe	Kirjoittaa asennuksen tietokantoihin tietoa ulkopuolisista arkistoista mihin asennus on yhdistetty.
Melt.exe	Muuttaa / kääntää MSM- tiedoston komponenttiryhmäksi WiX- lähdetiedostossa (.wxs)
Torch.exe	Suorittaa muutoksia kääntäessä (.wixmst tai .mst) tiedostot XML- lähtötiedostoihin (.wixout tai wixpdb) tai MSI- tiedostoihin.
Smoke.exe	Suorittaa MSI- ja MSM- tietojen validointia / kelpoisuustarkistusta.
Pyro.exe	Luo korjaustiedoston (.msp) tiedostoista (.wixmsp ja .wixmst).
WixCop.exe	Valvoo Wix standarttia. Muuntaa vanhan WiX version tiedostot uuden version mukaiseksi.
WixUnit.exe	Suorittaa WiX XML- tiedostojen ja lähtötiedoston validointia.

Taulukko 2. Lista WiX- työkaluista, sekä niiden kuvaukset. (Nick Ramirez. 2010. WIX Developer's Guide to Windows Installer XML. Birmingham. Packt Publishing. sivu 10).

4.1 Rungon Elementit

Jokaisen WiX- skeeman runko koostuu WIX XML- julistuksesta ja WiX- elementistä, mikä on koodin juuri elementti (Koodi 1). Kaikki muut elementit tulee sijoittaa WiX- elementin sisään.

```
<?xml version="1.0" encoding="UTF-8"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi">
</Wix>
```

Koodi 1. Esillä kaikista perus- ja triviaalitiedoston koodi, mikä sisältää yhden elementin ja jossa on linkki WiX- tiedoston prototyyppiin.

4.1.1 Tuote- elementti

Ensimmäisenä koodin runkoon on hyvä lisätä Product- elementti. Elementti kuvaa asennuksen tarjoamia tuotteita. Elementin sisään tulevat attribuutit, joiden arvot kertovat loppukäyttäjälle sekä tietokoneelle sovelluksesta ja helpottavat sen yksilöimistä. Ne ovat: Id-, Name-, Language-, Version-, Manufacturer- ja UpgradeCode- attribuutit. Attribuutti Id:in arvo on GUID:iksi kutsuttu numero. Jokaisella jaettavalla sovelluksella sekä lisäkomponentilla pitäisi olla GUID. GUID toimii sovelluksille sekä niiden komponenteille globaalisesti yksilöivällä tunnisteella Windows järjestelmärekisterissä. GUID antaa mahdollisuuden Windows Installerille valvoa tietokoneelle tehdyt asennukset, sekä estää sovelluksien keskinäistä sekaannusta. GUID määrittämisen voidaan käyttää Windows PowerShell:ia, erilaisia ilmaisia GUID:n luovia ohjelmia tai Visual Studio:n tarjoamaa välinettä. Visual Studio:n Creat GUID välinen avulla saa generoitua nopeasti GUID arvoja erimuotoisena. Name eli Nimi- attribuuttiin arvoksi tulee jaettavissa olevan sovelluksen nimi. Id- attribuuttiin ja sen GUID arvon ansiosta, jos asennuksen jälkeen tietokoneesta löytyisi kaksi erilaista mutta samannimistä sovellusta, tietokone kumminkin erottaisi ne toisistaan. Language- attribuutti määrittelee tuotteen lokalisoinnin LCID- kielitunnisteiden avulla. Suomen kielen arvo on 1035, Brittienglannin arvo on 2057 tai Amerikanenglannin arvo on 1033. Lisää tietoa eri kielten kielitunnisteista saa esimerkiksi osoitteesta http://msdn.microsoft.com/en-us/goglobal/bb964664. Versio- attribuutti antaa tiedon tuotteen versiosta. Manufacturer- attribuuttiin tulee tuotteen valmistajan kuvaus. Viimeinen attribuutti UpgradeCode ei ole pakollinen, mutta sen käyttö on suositeltava, joten sovelluspäivityksien teko onnistuisi. Sen arvoksi tulisi GUID- luku.

4.1.2 Pakkaus- elementti

Package eli pakkaus- elementti käsittelee itse asennusohjelman tietoja. Elementissä on vain yksi pakollinen attribuutti: Compressed. Sitä ja muita oleellisia attribuuttia käydään läpi alatekstissä. InstallerVersion- attribuutti määrittelee asennusohjelman yhteensopivuutta mahdollisen asennetun Windows Installer:in version kanssa. Attribuutin arvoksi syötetään vanhimman tuetun version vastaava kolminumeroinen luku. Nämä luvut saadaan kertomalla Windows Installer:in pääversion numero sadalla ja jos on olemassa vielä aliversion numero, se lisätään saadun luvun loppuun. Eli jos Windows Installer:in versio on 4.5, InstallerVersio- attribuutti arvoksi tulee 405. Compressed- attribuutti määrittelee tiedoston pakkausta. Sen voi joko sallia tai kieltää. Tarkemmin asia käydään läpi Media- elementissä. Manufacturer- attribuutti voidaan vapaaehtoisesti lisätä. Se on Product- elementin samannimisen attribuutin kaltainen ja siihen tulee tieto asennusohjelman tekijästä. Voidaan myös lisätä Description- attribuutti, joka antaa asennuksella kuvauksen.

4.1.3 Media- elementti

Seuraava elementti on Media. Jokaisella asennuspaketilla pitäisi olla ainakin yksi Media elementti. Media- elementti kuvaa arkistotiedostoa, jossa säilyy kaikki asennuspaketin sisältö. Isolla ohjelmalla voi olla useita arkistotiedostoja, joista jokainen voi olla eri fyysisillä tallennuslevyillä. Tiedot arkistoidaan Cabinet- tiedostoon, jonka pääte on .cab. Tiedostot ovat MSI-tiedoston sisällä näkymättömissä. Ne voidaan erottaa MSI- tiedostosta ja laittaa näkyviin lisäämällä EmbedCab- attribuutti, jonka arvoksi tulee ”no”. Jaettaessa asennusta fyysisillä tallennuslevyillä on tärkeää, että asennusvaiheet menevät oikein ja ohjelma asentuu onnistuneesti. Siihen tarvitaan Property- elementti sekä attribuutit: DiskPrompt ja VolumeLabel. Niiden tehtävä on valvoa ja panna täytäntöön eri tallennuslevyjen vaihto. Property- elementti hoitaa asennuslevyjen vaihtokehotusilmoituksia. Se käyttää Id arvona DiskPrompt- attribuutteja, jotka ovat jokaisen Media- elementin sisällä ja määräävät levyjen järjestystä. VolumeLabel- attribuutti nimeää levyn (Koodi 2).

```
<?xml version="1.0" encoding="UTF-8"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi">
    <Product Id="857d0aaa-6ae4-48c3-b96f-cbd08ea5630b"
        Name="TestiSovellus" Language="1035"  Version="1.0.0.0"  Manufacturer="TestiYritys"
        UpgradeCode="ca7564ff-0269-468b-ae67-c240f1b28034">
    <Package InstallerVersion="200" Compressed="yes"
            Manufacturer="Asennusohjelmantekijä "
            Description="Asennusohjelma TestiSovellusta varteen"/>
    <!-- Tallenusarkistojen järjestyksen valvominen -->
    <Property Id="DiskPrompt"
        Value="TestiSovellus – [1]" />
    <!-- Ensimmäinen ohjelman sisältävä tallennusarkisto -->
    <Media Id="1" Cabinet="testisovellus1.cab" EmbedCab="yes"
            DiskPrompt="Levy 1"  VolumeLabel=" Levy1" />
    <!—Toinen ohjelman sisältävä tallenusarkisto -->
    <Media Id="2" Cabinet="testisovellus2.cab" EmbedCab="yes"
        DiskPrompt="Levy 2"  VolumeLabel=" Levy2" />
    </Product>
</Wix>
```

Koodi 2. Kahdelle arkistotiedostolle jaettuun asennusohjelman koodirungon esimerkki.

4.2 Hakemistot

Hakemisto- elementti määrittää paikan, mihin jaettavat tiedostot tulee sijoittaa loppukäyttäjän tietokoneessa. Ensimmäiseksi on luotava Directory- juurielementti, jonka Id- arvo pitäisi olla ”TARGETDIR” ja Nimi- arvo - ”SourceDir”. Sen jälkeen käytetyt Hakemisto-elementit on laitettava tämän juurielementin sisään (Koodi 3, kohta A). Oletuksena TARGETDIR- arvoksi Windows Installer määrittää paikallisen kovalevyn, jossa on enemmän vapaata tilaa. Useimmin se on C: asema. Yleisille koodissa käytetyille hakemisto-osoitteille on olemassa lyhenteitä. Listan niistä voi katsoa Microsoftin sivuilla: http://msdn2.microsoft.com/en-us/library/aa370905%28VS.85%29.aspx. Esimerkkinä, ProgramFilesFolder tarkoittaa yleensä C:\Program Files osoitetta, joka on myös tiedostojen käytetyin sijoituspaikka. Siihen lisätään kaikki sisältö jaettavan tuotteen nimen mukaisen alakansion alle.

4.3 Komponentti- elementit

Hakemistojen selvittämisen jälkeen voidaan siirtyä asennettavan sisällön määrittämiseen, joka sijoittuu Komponentti (Component)- elementteihin. On suositeltavaa luoda koodin jokaiselle asennettavalle tiedostolle oma Komponentti- elementti. Komponentit voidaan laittaa joko Hakemisto (Directory) - elementin sisään tai sijoittaa ne erikseen koodissa. Silloin käytetään DirectoryRef Id- viittausta, jonka arvoksi laitetaan viitatun Hakemisto- elementin Id- arvo (Koodi 3 kohta B). Sijoittaminen erikseen komponenttiin voisi auttaa rakenteen purkua pitkässä koodissa. Jokaisella Komponentti- elementillä on pakolliset attribuutit: Id ja GUID, joiden arvot ovat ainutlaatuiset asennuspakettikoodin sisällä.

Komponentti- elementin sisään tulee Tiedosto (File)- elementti. Tiedosto- elementillä on myös oma Id- attribuutti. Attribuutti arvot: Name ja Source kertovat tiedoston nimen, jonka se saa asennuksen jälkeen sekä sijainnin, josta tiedosto löytyy ennen asennusta. Tiedosto-elementti sisältää Disk Id- attribuutin, joka viittaa Media- elementtiin ja kertoo missä arkistossa tiedosto sijaitsee. KeyPath- attribuutti vastaa siitä, että tiedoston puuttuminen huomataan mahdollisessa korjauksessa ja sen jälkeen asennetaan uudelleen. Sen arvoksi on laitettava ”yes” (Koodi 3, kohta C).

Kuvakkeet voivat sijoittua joko erikseen tai Tiedosto- elementin sisään. Jokaisella kuvakkeella on: Id-, Name-, Description-, WorkingDirectory-, Directory- ja Icon- attribuutti. Kolmelle ensimmäiselle attribuutille arvot ovat: yksilöivä tunnus, nimi ja kuvaus. Directory- attribuuttiarvoksi laitetaan osoitepolku, jossa kuvake tulee olemaan ja WorkingDirectory- arvoksi – osoitepolku, mihin kuvake viittaa. Ennen Icon- attribuutin arvon täyttämistä, siirretään Directory- tai DirectoryRef- elementtien ulkopuolelle ja tehdään siihen Icon- elementti (Koodi 3,

kohta D). Icon elementtiin määritetään Id- attribuutin arvo ja SourceFile- arvo, joka on tulevan kuvakkeen kuvan osoite. Kun elementti on luotu, voidaan siirtyä takaisin Icon- attribuuttiin ja laittaa sen arvoksi Icon- elementin Id-arvo, jotta syntyisi viittaus siihen. Advertised-attribuutin ”yes”- arvo on laitettava niiden Kuvakkeiden kohdalla, jotka viittaavat tiedostoihin jotka ovat asennettavissa pyydättäessä tai ulkoisesta asemasta.

4.4 Vaihtoehto- elementit

Feature- elementin lähin englanninkielinen käännös olisi Vaihtoehto- elementti. WiX:in kielilokalisoinnissa käytetään Ominaisuus nimeä, mutta Vaihtoehto on oikeampi nimeke elementin toiminnan kannalta. Se käsittää jaossa olevia tuotteita, jotka loppukäyttäjä saa valita oman tietokoneeseensa asennettavaksi asennuskäyttöliittymäikkunasta asennusohjelman toimiessa. Yksi Vaihtoehto- elementti voi koostua eritasoisista Vaihtoehto- elementtien kokonaisuuksista. Niiden sisällä viitataan mahdollisiin asennettaviin komponentteihin ja sitä kautta niiden alla oleviin tiedostoihin. Vaihtoehto- elementillä on olemassa paljon erilaisia attribuutteja. Yleisiä ovat: Id, ConfigurableDirectory, Description ja Title. Id-attribuutin arvo toimii yksilöivänä tunnisteena. ConfigurableDirectory- attribuutin arvo antaa mahdollisuuden loppukäyttäjälle valita asennuspaikan. Level- attribuutti määrää niin sanotun asennuksen tason. Level arvoksi tule laittaa numeroita. Jos arvoksi laitetaan nolla, alla olevaa komponenttia ei asenneta. Numero yksi tarkoittaa asennusta oletustilassa. Kaikki muut ykkösen ylittävät arvot mahdollistavat niiden alla olevien komponenttien asennuksen. Oletuksena asennus on kytketty pois. Title- ja Description- attribuuttien arvoista tulevat otsikko ja kuvaus mahdollisesta asennettavasta tuotteesta, jotka ovat loppukäyttäjälle näkyvillä asennuskäyttöliittymäikkunasta. Viittaukset asennettaviin komponentteihin, jotka tulevat Vaihtoehto- elementtiin alle toimivat ComponentRef- elementin avulla, jonka Id- arvona on sama Id- arvo kuin asennettavilla komponenteilla (Koodi 3, kohta E).

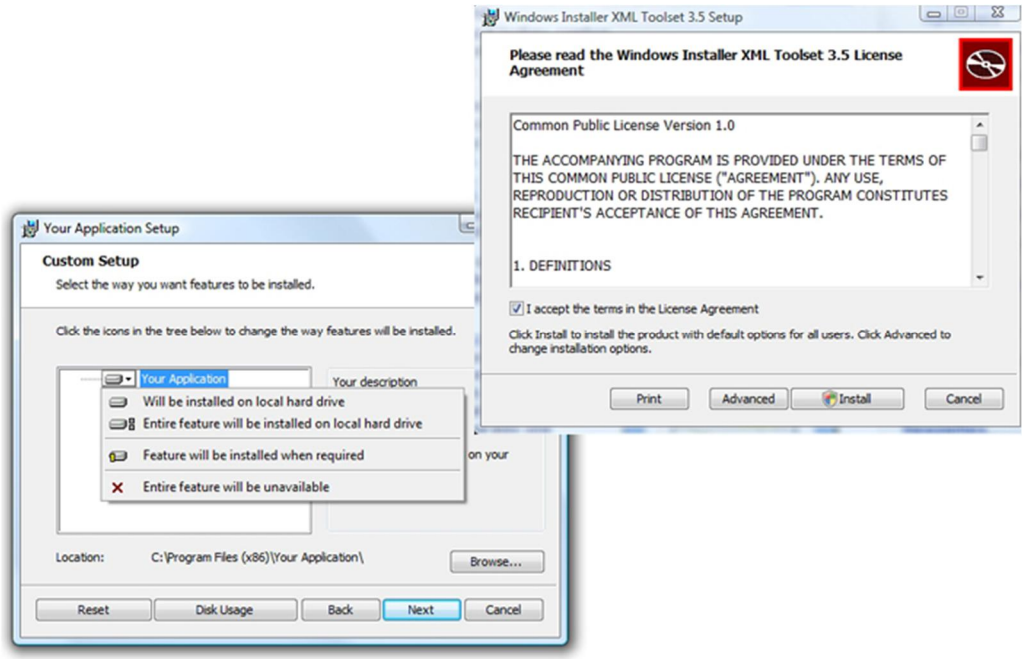
Käytyä läpi Vaihtoehto- elementin ja kaikki sen ennen mainitut elementit voidaan sanoa, että kaikki WiX- skeeman koodin alkeet on opittu. Noilla tiedoilla, koottaessa kaikki yhteen ja ottamalla sen esimerkkinä miten pitäisi toimia, pystyy tekemään toimivan ja perustiedot sisältävän MSI- asennuspaketin (Koodi 3). Kaikki siinä käytetyt elementit ovat läsnä jokaisessa WiX- teknologiaa käyttäen tehdyssä asennusohjelmassa. Haluttaessa ja tarpeiden mukaan koodiin voi lisätä vielä paljon lisäkohtia, jotka eivät tulleet mainituksi tässä, mutta ovat olemassa ja ovat osana WiX- teknologiaa.

```
<?xml version="1.0" encoding="UTF-8"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi">
    <Product Id="857d0aaa-6ae4-48c3-b96f-cbd08ea5630b"
        Name="TestiSovellus" Language="1035"
        Version="1.0.0.0"  Manufacturer="TestiYrityksenNimi"
        UpgradeCode="ca7564ff-0269-468b-ae67-c240f1b28034">
    <Package InstallerVersion="200" Compressed="yes"
        Manufacturer="AsennusohjelmanTekijä "
        Description="Asennusohjelma TestiSovellusta varteen"/>
    <Media Id="1" Cabinet="testisovellus1.cab" EmbedCab="yes" />
    <!-- Hakemistoluettelo (Kohta A) -->
    <Directory Id="TARGETDIR" Name="SourceDir">
        <!-- Hakemisto ProgramFiles\TestiYrityksenNimi\TestiSovellus -->
        <Directory Id="ProgramFilesFolder">
            <Directory Id="TestiYrityksenKansio" Name="TestiYrityksenNimi" >
                <Directory Id="TestiAsennusKansio" Name="TestiSovellus"/>
            </Directory>
        </Directory>
        <!-- Hakemisto Työpöytä>
        <Directory Id="DesktopFolder"/>
    </Directory>
    <!-- Jaettavissa olevan sovelluksen sisaltö (Kohta B) -->
    <DirectoryRef Id="TestiAsennusKansio">
        <!-- Varsinainen asennettava tiedosto (Kohta C) -->
        <Component Id="ItseTestiOhjelma" Guid="1603164F-1BC8-4350-A005-A82487610B24">
            <File Id="TestiSovellusTiedosto" Source="Sovellus.exe" KeyPath="yes">
                <!-- Työpöytakuvake sovelluksen käynnistämistä varteen -->
                <Shortcut Id="TyopoytaKuvake" Name="TestiSovellus"
                WorkingDirectory="TestiAsennusKansio" Directory="DesktopFolder"
                Icon="Kuvake.ico"></Shortcut>
            </File>
        </Component>
    </DirectoryRef>
    <!-- Kuvakkeen kuva (Kohta D) -->
    <Icon Id="Kuvake.ico" SourceFile="kuvake1.ico" />
    <!-- Asennuksen Vaihtoehto- elementti  (Kohta E) -->
    <Feature Id="TestiTaysAsennus" Title="TestiAsennus"
        Description="Asentaa kaikki mainitut tiedostot" Level="1">
          <ComponentRef Id="ItseTestiOhjelma"/>
    </Feature>
    </Product>
</Wix>
```

Koodi 3. Esimerkki perustiedot sisältävästä asennusohjelman koodista, sekä tekstissä käytettyjen kohteiden selitys.

4.5 Käyttöliittymät

WiX mahdollistaa UI:in (User interface) eli käyttöliittymien käytön asennusohjelmassa. Se ei ole kuitenkaan pakollista. Asennus voidaan suorittaa ilman käyttöliittymän ilmestymistä eli hiljalleen ilman dialogi-ikkunoita. WiX- kokoonpanoon kuuluu valmiita dialogi-ikkunoita sisältävät käyttöliittymäpaketit. Dialogi-ikkunoita voidaan rakentaa itse tai käyttää valmiita pohjia käyttöliittymäpaketeilta: WixUI_Advanced, WixUI_Mondo, WixUI_FeatureTree, WixUI_InstallDir, WixUI_Minimal. Käyttöliittymäpaketti voidaan valita riippuen siitä miten hienostunut asennusohjelma halutaan. WixUI_Minimal on minimaalisen määrän dialogi-ikkunoita sisältävä käyttöliittymäpaketti. Mukana on vain yksin dialogi-ikkuna, jossa on tila vain käyttäjälisenssille ja Asenna napille. WixUI_Advanced antaa mahdollisuuden valita kuinka monelle tietokoneen käyttäjälle ohjelma asennetaan. Lisäksi voidaan valita ohjelman sijaintikansio sekä tiettyjä ohjelman komponentteja asennettavaksi. WiXUI_FeatureTree sisältää melkein samoja ominaisuuksia kuin WixUI_Advanced. WixUI_InstallDir antaa käyttäjälle mahdollisuuden valita kohdehakemisto ilman räätälöityjä ominaisuuksia. WixUI_Mondo- käyttöliittymäpaketissa tulee valita yksi kolmesta asennustyypeistä. Typical- asennus eli asennus, jossa asennetaan yleisimmät tarvittavat ominaisuudet. Tätä yleensä suositellaan kaikille ohjelman tuleville peruskäyttäjille. Custom- asennuksella käyttäjille annetaan mahdollisuus valita halunsa mukaan ohjelman komponentteja (Kuva 5). Complete- asennus asentaa kaikki tarjolla olevat ohjelmistot.



Kuva 5. Asennusohjelman käyttöliittymän dialogi-ikkunoiden esimerkki.

Käyttöliittymän (UI:n) toimintaan Visual Studio:ssa pitää lisätä viittaus kirjastoon, jossa tiedot käyttöliittymistä säilytetään. Viittaus lisätään Referenses kohdasta ja valitsemalla sen ainut vaihtoehto Add References. Näkyviin tulee C:\Program Files\Windows Installer XML v<Version numero>\bin kansion sisältö, josta valitaan WixUIExtrnsuion.dll tiedosto. Sen jälkeen lisätään varsinaiseen koodiin Tuote- elementin alle koodipätkä, jossa viitataan tarvittavaan käyttöliittymän nimeen (Koodi 4) sekä lisätään asetukset dialogi-ikkunoihin, joista esimerkit käydään läpi myöhemmin.

```
<UIRef Id="WixUI_Minimal" />
```

Koodi 4. Käyttöliittymään viittaaminen koodissa.

5 Käytännön esimerkki asennusohjelmasta

Perusteiden läpi käynnin jälkeen voidaan testata omia tietoja ja taitoja tekemällä toiminnallinen asennusohjelma. Samalla tutustutaan erilaisiin lisäominaisuuksiin. Ennen asennusohjelman tekoa selvitetään, mitä tuotetta varten asennusohjelma tehdään. Seuraavaksi selvitetään mahdolliset käyttäjät ja käyttäjäryhmät, jotka saattavat käyttää kyseistä tuotetta ja sen mukana tulevaa asennusohjelmaa. On otettava huomioon käyttäjien tietokoneen käyttötaidot, mielipiteet ja odotukset tuotteesta sekä mitkä ovat yleisiä trendejä ja sääntöjä tuotetta edustavalla alalla. Tämä kaikki auttaa paljon asennusohjelman rakenteen ja visuaalisen ilmeen suunnittelussa ja tekemisessä. Tavoitteena on miellyttää tulevaa tuotteen käyttäjää eikä pelottaa häntä jo asennuksen aikana ennen tuotteen käyttämistä.

5.1 Lähtökohdat

Määritetään esimerkkiasennusohjelman tuote. Otetaan jaettavaksi tuotteeksi keksityn yrityksen läpimurtotuote. Tässä tapauksessa on tärkeintä näyttää asennusohjelman luomisen tekninen prosessi. Syventäminen sen valmistaviin toimenpiteisiin on suurelta osin turhauttavaa. Jos esimerkkinä ei olisi keksitty tapaus vaan oikea yritys ja tuote, valmistelu- ja suunnitteluvaihe olisi pitkä ja huolellinen sekä perustuisi tutkimuksiin ja muihin luotettaviin tietoihin.

Oletetaan, että kyseessä on pieni noin 5-7 henkeä työllistyvä IT-alan yritys. Annetaan yritykselle nimi - Huippu Koodarit Oy. Läpimurtotuotteena heillä olisi graafista sisältöä muokkaava ja tuottava sovellus, minkä nimi olisi Isku. Nimet ovat täysin keksittyjä, joten kaikki yhteensopivuudet ovat satunnaisia. Sovellus olisi tarkoitettu sekä ammattigraafikoille että kuvankäsittelyä harrastaville ja aloittelijoille. Tuotteen mukana tulisivat vapaaehtoisesti asennettavat dokumentaatiot, jotka auttavat loppukäyttäjiä sovelluksen oppimisessa. Pienellä yrityksellä ei ole varaa ostaa kallista ohjelmistoa. Heidän valintansa on selvä, jos yritys oikeasti haluaa hy-

vän, monipuolisen ja visuaalisesti miellyttävän asennusohjelman omalle tuotteelle. Se on WiX työkalupaketti.

5.2 Asennusohjelman teko

Tämän asennusohjelman esimerkin tekoon tarvittavat työkalut: sovelluskehitysalusta, tässä tapauksessa Visual Studio 2010, WiX- työkalupaketti, sekä Orca- ohjelma. Oletetaan, että kaikki ne on asennettu käyttövalmiiksi. Työkalujen lisäksi tarvitaan asennusohjelman asennettava sisältö sekä asennusohjelman mahdolliset osat: lisenssin sisältävä tekstitiedosto, kuvakkeet ja muut ulkonäköön liittyvät tiedostot. Esimerkkiohjelmassa käytettävät tiedostot: isku.exe, lisenssi.rtf, iskukuva.ico, iskufaq.ico, iskuinfo.ico, kayttoohje.rtf. Lisää niistä kerrotaan niiden koodin lisäämisen aikana.

Avataan Visual Studio- sovelluskehitysalusta ja luodaan uusi projekti. Valitaan esille tulleesta sapluunavalikosta Windows Installer XML ja Setup Project- malliprojekti. Projektin nimeksi laitetaan tuotteen nimi eli Isku. Lisäksi valitaan projektikansion sijainti. Luotuun Isku.wxs lähdetiedostoon tulee valmis mallikoodi vähäisten englanninkielisten selityksien kanssa, jonka avulla voidaan lähteä rakentamaan asennusohjelmaa.

5.2.1 Perustiedot koodiin

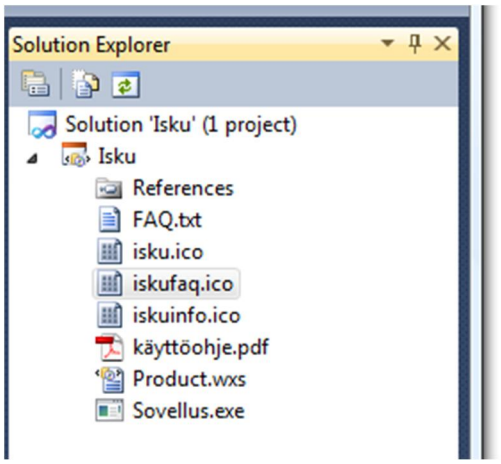
Valmiiksi luotua Isku.wxs- tiedoston koodia muokataan käyttäen opittuja tietoja. Kaikki WIX-skeeman runkoon kuuluvat elementit on jo luotu automaattisesti, joten siirrytään heti niiden täyttämiseen. Ensimmäiseksi otetaan Tuote-elementti, jossa muokataan tuotteen tietoja. Manufacturer- kohtaan laitetaan jaettavan tuotteen valmistajayrityksen nimi sekä Kieli- attribuutin arvoksi suomen kieltä vastaava luku 1035. Pakkaus- elementtiin voidaan lisätä asennuspaketin tekijä Manufacturer- attribuutti- arvona. Media- elementtiin ei kosketa, koska tuotteemme jaetaan yhtenäisenä. Asennuspaketin sisältö jakaantuu loppukäyttäjän tietokoneessa kolmeen eri hakemistoon. Käytetään tyypillisiä hakemistoja ohjelmien asennuksessa: Program Files kansio, Työpöytä, sekä Käynnistä\Ohjelmat valikko. Lisäksi ProgramFiles ja Käynnistä\Ohjelmat hakemistoihin luodaan alahakemistoja (Koodi 5).

```
<?xml version="1.0" encoding="UTF-8"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi">
    <Product Id="17a07542-7e9e-40ea-a45a-dae1cbc04b34" Name="Isku" Language="1035"
        Version="1.0.0.0" Manufacturer="Huippu Koodarit Oy"
        UpgradeCode="19327428-c3f5-45dc-b9fc-aea1259a2141">
    <Package InstallerVersion="200" Manufacturer="Huippu Koodari Pekka" Compressed="yes"/>
    <Media Id="1" Cabinet="media1.cab" EmbedCab="yes" />
    <!-- Asennushakemistot -->
    <Directory Id="TARGETDIR" Name="SourceDir">
        <!-- Hakemisto ProgramFiles\Huippu Koodarit Oy\Isku   -->
        <Directory Id="ProgramFilesFolder">
            <Directory Id="YrityksenKansio" Name="Huippu Koodarit Oy">
                <Directory Id="ASENNUSKANSIO" Name="Isku">
                    <!-- Alahakemisto ProgramFiles\Huippu Koodarit Oy\Isku\Dokumentatiot  -->
                    <Directory Id="FAQKansio" Name="Dokumentatiot"/>
                </Directory>
            </Directory>
        </Directory>
        <!-- Alahakemisto Käynnistä\Ohjelmat\Isku -->
        <Directory Id="ProgramMenuFolder" Name="Programs">
            <Directory Id="ProgramMenuDir" Name="Isku"/>
        </Directory>
        <!-- Työpöytä  -->
        <Directory Id="DesktopFolder" Name="Desktop"/>
    </Directory>
```

Koodi 5. Isku asennusohjelman koodinpätkä, joka sisältää rungon elementit ja hakemistot.

5.2.2 Asennettavat tiedostot

Lisätään Visual Studio- projektiin jaettava sisältö helpottamaan mahdollisia viittauksia koodissa. Valitaan Visual Studion ylämenusta Project ja Add Existing Item kohta ja sieltä kaikki tarvittavat tiedostot. Tiedostot ilmestyvät oletuksena oikealla olevaan Solution Explorer- ikkunaan projektin nimen alle (Kuva 6). Kuvasta nähdään, että asennusohjelma asentaa varsinaisen sovelluksen lisäksi käyttöoppaan ja FAQ.txt- tiedoston, joka sisältää usein kysyttyjä kysymyksiä. Sen lisäksi asennetaan myös linkkitiedosto Huippu Koodarit Oy:n kotisivuille, mutta linkkitiedostoa ei lisätty projektiin, koska tämä toteutetaan käyttäen omaa teknologiaa.



Kuva 6. Isku asennusohjelman asennettava sisältö (Visual Studio näkymä).

Ohjelman sovellus laitetaan ProgramFiles- kansion asennuksen aikana luotuihin alakansioihin Huippu Koodarit Oy ja Isku alle. Alahakemistoon, jonka Id- arvo on FAQKansio, sijoitetaan tiedostot: Käyttöopas.pdf ja FAQ.txt. Jokaiselle tiedostolle tulee oma etukäteen valmistettu kuvakkeen kuva. Sovelluksen pikakuvake menee loppukäyttäjän Työpöydälle. Yleisiä kysymyksiä tuotteesta sisältävää FAQ.txt- tiedostoa ei pakata MSI- tiedostoon vaan jaetaan sen mukana, koska jo ennen tuotteen asennusta FAQ.txt- tiedosto voi olla hyödyllinen tuotteen loppukäyttäjille. Esimerkiksi se voisi pitää sisällään vastauksia tuotteen asennuksen mahdollisiin ongelmiin. Tämä toteutetaan lisäämällä koodissa tiedoston sisältämään File- elementtiin Compressed- attribuutti arvona ”no”. Lisäksi laitetaan samaa File- elementtiä sisältävään Komponenti- elementtiin Location- attribuutti arvona ”either”. Se liittyy tiedoston asennusvaihtoehtoihin, joten siihen vielä palataan myöhemmin (Koodi 7).

```
<!-- Käyttössä olevat ikonit -->
<Icon Id="iskukuvake.ico" SourceFile="isku4.ico"/>
<Icon Id="opaskuvake.ico" SourceFile="isku6.ico"/>
<Icon Id="FAQkuvake.ico" SourceFile="isku5.ico"/>
<!-- Varsinainen asennettava sisältö  -->
<DirectoryRef Id="ASENNUSKANSIO">
    <Component Id="IskuSovellus" Guid="1603164F-1BC8-4350-A005-A82487610B24">
        <File Id="IskuTiedosto" Source="Sovellus.exe" KeyPath="yes">
            <!-- Sovelluksen kuvakkeet käynnistämistä varteen -->
            <Shortcut Id="PikakaynnistysKuvake" Name="Isku käynnistys"
                WorkingDirectory="Isku" Directory="ProgramMenuDir"
                Advertise="yes" Icon="iskukuvake .ico"/>
            <Shortcut Id="TyopoytaKuvake" Name="Isku" WorkingDirectory="Isku"
                Directory="DesktopFolder" Advertise="yes"  Icon="iskukuvake.ico"/>
        </File>
    </Component>
</DirectoryRef>
<!-- Dokumentatiot hakemiston sisältö -->
<DirectoryRef Id="FAQKansio">
    <Component Id="KayttoOpas" Guid="389A9E01-89C8-41F3-970E-269560D6678B">
        <File Id="KayttoOpastiedosto" Name="käyttöopas.pdf"
        Source="käyttöopas.pdf" KeyPath="yes">
            <Shortcut Id="Opaskuvake" Name="KäyttöOpas" WorkingDirectory="FAQKansio"
                 Directory="ProgramMenuDir" Advertise="yes" Icon="opaskuvake.ico" />
        </File>
    </Component>
    <Component Id="FAQ" Guid="AF60B531-22FC-42C3-90DA-2F7A07F2155A" Location="either">
            <File Id="FAQtiedosto" Name="FAQ.txt" Source="FAQ.txt"
                KeyPath="yes" Compressed="no">
                    <Shortcut Id="FAQtiedostokuvake" Name="FAQ" WorkingDirectory="FAQKansio"
                    Directory="ProgramMenuDir" Advertise="yes" Icon="FAQkuvake.ico"/>
            </File>
    </Component>
</DirectoryRef>
```

Koodi 7. Isku- asennusohjelman avulla asennettavat tiedostot koodissa.

5.2.3 Poistaminen

On hyvä tapa luoda Käynnistä\Ohjelmat- valikkoon tuotteen nimisen kansion alle ohjelman asennettavan sisällön poistamisen mahdollistava kuvake. Kuvake sijoitetaan Komponentti-elementtiin sisään, joka menee loppukäyttäjäkoneella ProgramMenuDir- hakemistoon eli Käynnistä- valikkoon. Komponentti koostuu kolmesta elementtistä. Kuvake- (Shortcut) elementti viittaa omana Target- arvona SystemFolder- hakemistoon eli järjestelmäkansioon ja

siinä olevaan msiexec.exe tiedostoon eli Windows Installeriin ja lähettää siihen Arguments-attribuutin ProductCode- arvon, mikä on tuotteemme GUID. SystemFolder sanojen keskellä sijaitsevat numerot riippuvat tuotteen käyttöön tarkoitetusta järjestelmätyypistä. Jos jaossa oleva tuote on tarkoitettu molempiin järjestelmätyyppeihin, on laitettava järjestelmäkansion nimeen numero 64. Toinen Komponentin sisältämä elementti poistaa kansion johon viittaa sen Id- arvo. Kansion poistamiseksi sen attribuutti On- arvo tulee olemaan ”uninstall”. Viimeinen RegisteryValue- elementti valvoo tuotteen olemassaoloa järjestelmässä. Root- attribuutin arvo on rekisteriavain, joka tarkistaa nykyisen käyttäjäprofiilin ja onko tuotetta asennettu sen alle (Koodi 8).

```
<!-- Poisto kuvake -->
<DirectoryRef Id="ProgramMenuDir">
    <Component Id="Poistaminen" Guid="4BE0BB91-5568-471C-AE0E-4DC582A6D640">
        <Shortcut Id="SovelluksenPoisto" Name="Isku Asennuksen poisto"
            Target="[Systen64Folder]msiexec.exe" Arguments="/x [ProductCode]"/>
        <RemoveFolder Id="ProgramMenuDir" On="uninstall"/>
        <RegistryValue Root="HKCU" Key="Software\[Manufacturer]\[ProductName]"
            Type="string" Value="1" KeyPath="yes"/>
    </Component>
</DirectoryRef>
```

Koodi 8. Poistokuvakkeen luonti.

5.2.4 Hyperlinkkikuvake

Jaettavissa oleviin tiedostoihin lisätään tiedosto, jonka asentamista loppukäyttäjä ei voi valita. Tässä tapauksessa kyseessä on hyperlinkkikuvake, joka lähettää sen avaajan tiettyyn osoitteeseen. Se asentuu päätiedoston mukana ja sen tarkoituksena on mainostaa tuotteen valmistajia ja niiden kotisivuja. WiX- työkalupaketin laajennuskirjasto WixUtilExtension antaa mahdollisuuden luoda sellaisen hyperlinkkikuvakkeen. Se lisätään Visual Studio- projektiin Add References- toiminnolla. Koodissa Wix- elementtin sisään lisätään viittaus kirjastoon: xmlns:util="http://schemas.microsoft.com/wix/UtilExtension". Näiden toimintojen jälkeen voidaan siirtyä kuvakkeen luomiseen. Directory Ref- elementtiin ja Asennuskansioon hakemistoviittauksen alle IskuTiedosto- komponentin lisäksi luodaan toinen komponentti. Id- komponentti nimetään ”Valmistajankotisivut” ja lisätään sille oma yksilöivä Guid. Komponenttin sisään lisätään util:InternetShortcut- elementti täytettyjen Id-, Name- ja Target- attribuuttien arvojen kanssa. Target- attribuutti arvona lisätään haluttu internetosoite, jonka jälkeen saamme toimivan hyperlinkkikuvakkeen (Koodi 9).

```
<!-- Hyperlinkki valmistajiin kotisivuihin -->
<Component Id="Valmistajankotisivut" Guid="BD85BCFB-5AD7-49D6-A70F-0098D2A0AC74">
    <util:InternetShortcut Id="Linkkikuvake" Name="Huippu Koodarit OY kotisivut"
        Target="http://www.google.com/"/>
</Component>
```

Koodi 9. Hyperlinkki- komponentin lisääminen.

5.2.5 Asennuksen vaihtoehdot

Feature eli Vaihtoehto- elementtipuu koostuu varsinaisesta ja vaihtoehtoisista asennussisältökokonaisuuksista. Varsinaiseen sisältöön kuuluvat itse tuotteen sovellus, sen mukana tuleva hyperlinkkikuvake ja poistamiskuvakkeen sisältävä komponentti. Kaikki nämä asentuvat tietokoneeseen välittömästi asennusohjelman toiminnan aikana.

Vaihtoehtoiseen kokonaisuuteen kuuluvat komponentit ovat asennettavissa myös toisilla tavoilla. Esimerkiksi joitakin tiedostoja tarvitaan vain tietyissä tilanteissa. Silloin voidaan käyttää asennusta pyydettäessä. Tämä tapa otetaan käyttöön käyttöoppaan sisältävän komponentin asennuksen kohdalla. OpasAsennuksen Vaihtoehto- elementin kolme viimeistä attribuuttia vastaavat siitä miten komponentti voidaan asentaa. AllowAdvertise- attribuutilla voisi olla kolme arvoa: ”yes”, ”no” ja ”system”. ”No” on oletus arvo, jolloin komponentti pyydättäessä ei asennu. Arvo ”yes” tarkoittaa sitä, että komponentti asentuu pyydättäessä, mutta kuitenkin on suositeltava käyttää ”system” arvoa. Se tarkoittaa, että asennus pyydättäessä on mahdollinen silloin kun tietokoneen järjestelmä tukee sitä. Komponentille pitää luoda itsenäinen kuvake, jolla on Advertise- attribuutti arvona ”yes”, jotta komponenttiin saataisiin yhteys ja sen jälkeen asennettaisiin tietokoneelle. Silloin kuvake asentuu tietokoneelle yksin ilman sen viittaamaa tiedostoa. Kun kuvaketta painetaan alkaa Windows Installer:in toiminta, joka tarkistaa tiedoston olemassaolon ja asentaa sen tarvittaessa.

Toinen asennustapa olisi asennus lähteestä, kuten ulkoisesta asemasta tai verkosta. Asennustavan toteutus on hyvin samankaltainen, mutta sillä on omia lisäkohtia. Tärkein attribuutti siinä on InstallDefault, jolla myös voi olla kolme eri arvoa:” local” eli paikallinen asennus, ”followParent”, joka toistaa ylemmän komponentin asennuksen ja ”source”, joka on asennus lähteestä. Jotta asennus toimisi oikein, asennettava tiedosto ei saa olla sulautettuna MSI-asennuspakettiin. Siksi lähteestä tulevaan asennettavan tiedoston komponenttiin lisätään Location- attribuutti arvona ”either” ja sen alla olevaan tiedosto elementtiin Compressed- attribuutti arvona ”no”. Tiedostoon viittaava kuvake pitää sisältää Advertise- attribuutin arvona ”yes”. Absent- attribuutti Vaihtoehto- elementeissä vastaa siitä, saako käyttäjä kytkeä vaih-

toehtokomponentin asennus pois asennuksesta käyttöliittymässä. Arvo ”allow” sallii käyttäjän poistaa sen. Jos arvo on ”disallow”, käyttöliittymä ei näytä poiskytkentä vaihtoehtoa (Koodi 10).

```
<!-- Asennuksen vaihtoehdot -->
<Feature Id="Asennus" Title="Asennus" Description="Isku Asennus"
    Display="expand" Level="1" ConfigurableDirectory="ASENNUSKANSIO"
    AllowAdvertise="no" Absent="disallow" InstallDefault="local">
    <!-- Varsinainen asennettava sisältö -->
    <Feature Id="OletusAsennus" Title="Isku Sovelluksen asennus" Level="1"
         AllowAdvertise="no" Absent="disallow" InstallDefault="local">
        <ComponentRef Id="IskuSovellus"/>
        <ComponentRef Id="Valmistajankotisivut"/>
        <ComponentRef Id="Poistaminen"/>
    </Feature>
    <!-- Vaihtoehtoiset asennukset -->
    <Feature Id="LisaAsennukset" Title="Lisäasennukset"
        Description="Asentaa lisäasennukset käyttäjän valinnan mukaan" Level="2"
         AllowAdvertise="no" Ab-sent="disallow" InstallDefault="source">
        <Feature Id="FAQAsennus" Title="FAQ tiedoston asennus"
            Description="FAQ tiedosto sisältä useimmin kysytyt kysymykset Isku sovelluksen liit-
            tyen" Level="2" AllowAdvertise="no" Absent="disallow" InstallDefault="source">
            <ComponentRef Id="FAQ"/>
        </Feature>
        <Feature Id="OpasAsennus" Title="Käyttöoppaan asennus"
             Descripti-on="Asentaa käyttöoppaan käyttäjän apuvälineksi" Level="2"
             AllowAdverti-se="system" Absent="allow" InstallDefault="local">
            <ComponentRef Id="KayttoOpas"/>
        </Feature>
    </Feature>
</Feature>
```

Koodi 10. Isku- esimerkkiasennusohjelman asennusvaihtoehtoja koodissa.

5.2.6 Käyttöliittymän luominen

Isku- asennusohjelmassa käytetään WixUI_FeatureTree- käyttöliittymää. Sen dialogi-ikkunoiden sisältävän pakettiviittauksen jälkeen, viitataan myös WixUI_ErrorProgressText-pakettiin, joka sisältää yleisiä varoituksia sisältäviä dialogi-ikkunoita. Nykyään jokainen tuote suojataan ja jaetaan jonkun lisenssin mukaisesti. Lisenssisopimuksen lisääminen asennusprosessiin ja sen hyväksyminen ennen varsinaista asennusta on yleistä tuotteiden jakamisessa. Lisenssi lisätään asennuksen käyttöliittymään käyttämällä WixVariable- muuttujaa, jonka Id on WixUILicenseRtf ja Value- arvo on lisenssitiedoston osoite. Tiedoston pitäisi olla .rtf- päätteineen tekstitiedosto. Käyttöliittymä voidaan kustomoida lisäämällä siihen visuaalisuutta.

Wix- teknologian avulla tehdyissä asennuskäyttöliittymissä on oletuksena omat Wix- banneri-kuvat ja kuvakkeet, joita pystyy halutessa vaihtamaan omiin käyttämällä samaa muuttujatekniikkaa (Koodi 11). Pitäisi kuitenkin tietää oikeat resoluutiot estääkseen kuvien epäesteettistä venymistä (Taulukko 3).

Muuttujan Nimi	Kuvaus
WixUIBannerBmp	Kuva sijoittuu kaikkiin dialogi-ikkunoiden yläreunaan, pois lukien ensimmäistä ja viimeistä (jos ne ovat käytössä), mitat 493 × 58 pikseliä.
WixUIDialogBmp	Taustakuva ensimmäisessä ja viimeisessä dialogi-ikkunassa, mitat 493 × 312 pikseliä.
WixUILicenseRtf	Lisenssiä sisältävän RTF tekstitiedoston osoite.

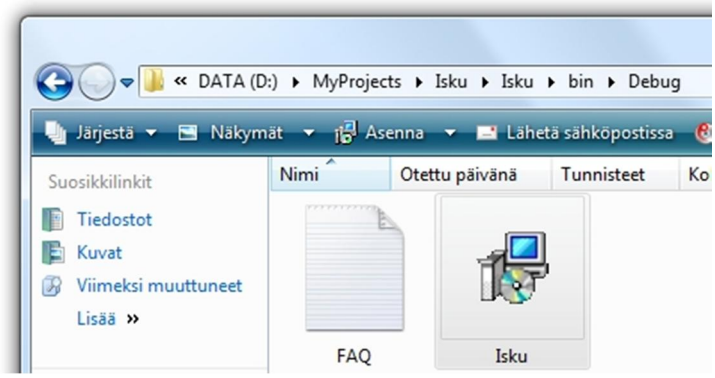
Taulukko 3. Asennusohjelman esimerkissä käytetyt muuttujat ja niiden kuvaus.

```
<!-- Asennuksen käyttöliittymä -->
<UI Id="IskunKayttoliityma">
    <UIRef Id="WixUI_FeatureTree"/>
    <UIRef Id="WixUI_ErrorProgressText"/>
</UI>
<!-- Kustomointi-->
<WixVariable Id="WixUIBannerBmp" Value="isku5.bmp" />
<WixVariable Id="WixUILicenseRtf" Value="lisenssi.rtf" />
<WixVariable Id="WixUIDialogBmp" Value="isku6.bmp"/>
```

Koodi 11. Isku- esimerkkiasennusohjelman käyttöliittymän kuvaus koodissa.

5.3 Valmiin asennusohjelman kokoonpano ja testaus

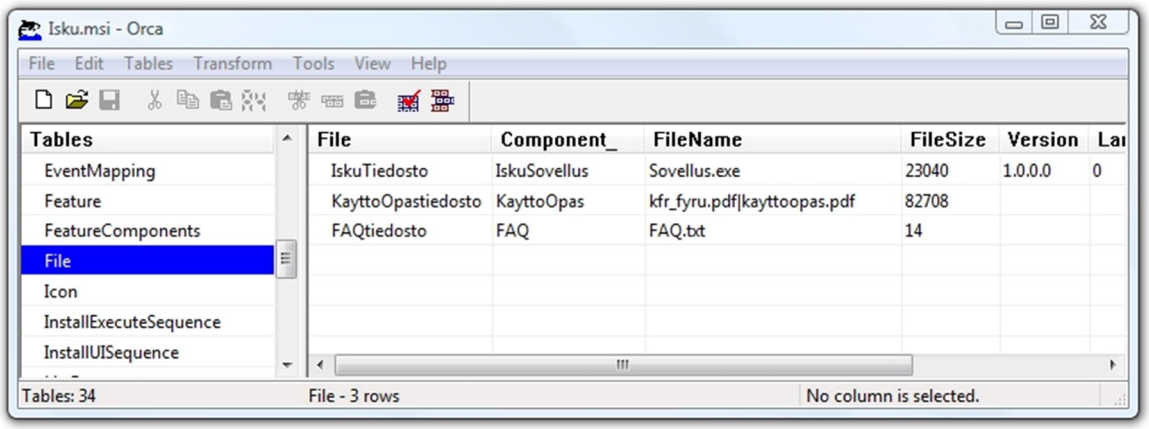
Kun koodi on kirjoitettu, kaikki asennuksessa jaettavat ja siihen liittyvät tiedostot lisätty projektiin sekä mainitut koodissa, voidaan koota kaikki asennusohjelman projektin sisältö yhdeksi asennuspaketiksi. Se tehdään valitsemalla sovelluskehitysalustan menusta Built- kohdan alavaihtoehto ”Built”, jonka jälkeen seuraa projektin nimi. Ennen siitä voidaan valita kokoamismuoto: Debug - virhetestausta varten tai Release - lopullista julkaisua varten. Jos jossain on tehty virhe tai jotain tärkeää on unohdettu, alaikkunaan ilmestyvät Error- viestit eikä kokoonpano onnistu. Jos kaikki on tehty oikein, lopputulokset löytyvät projektin luomisen vaiheessa määritetystä polusta ja siinä projektin nimisen kansion alakansioista: bin ja Debug tai Realese, riippuen sitä, mitä kokoamismuotoa käytettiin (Kuva 7). Tiedostojen löydyttyä voidaan olettaa, että asennuspaketti on tehty, mutta on hyvä vilkaista tiedoston sisältöä etukäteen ja testata asennuspakettia itse ennen sen jakamista eteenpäin muille.



Kuva 7. Asennusohjelmaprojektin lopputulos, Isku- asennuspaketti ja erikseen FAQ- tiedosto Debug- kansion alla.

5.3.1 Orca

Orca- ohjelma mahdollistaa MSI- asennuspakettien sisällön sekä rakenteen tutkimisen ilman niiden asennusta. Avattaessa Isku- asennuspaketti Orca- ohjelmalla voidaan tutkia taulukkomuotoisena: mitkä tiedot tallentuivat asennuspakettiin, mitkä tiedostot se pitää sisällään, mitä dialogi-ikkunoita se tulee käyttämään sekä paljon muuta (Kuva 8). Löydettäessä virheitä tai täydentämistä asennuspaketti voidaan korjata syöttämällä taulukkokenttiin halutut arvot. Pienet korjaukset ja lisäykset onnistuvat mainiosti Orca- ohjelman avulla. Isompiin muutoksiin suositellaan hyvää MSI- asennuspakettien tuntemusta tai paketin purkua Visual Studio:ssa WIX:in dark- työkalun avulla, koodin uudelleen täydentämistä ja takaisiin kokoamista yhteen light- työkalulla.

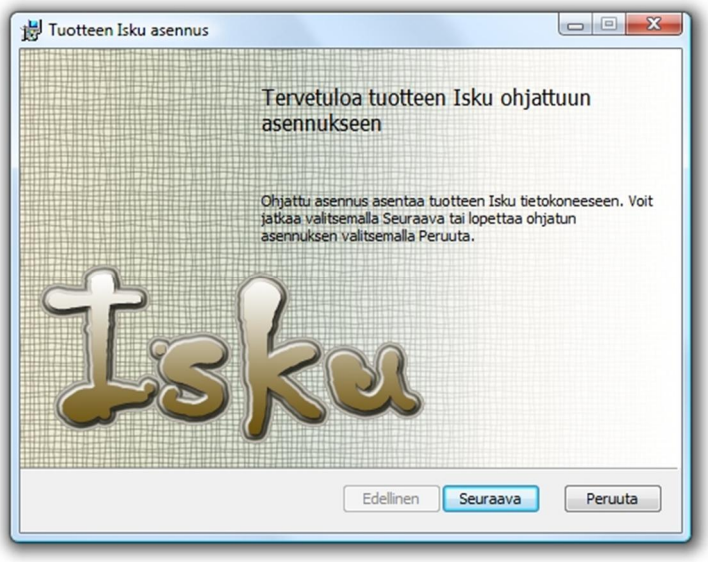


Kuva 8. Isku- asennuspaketin sisältö Orca- ohjelman avulla.

5.3.2 Asennus

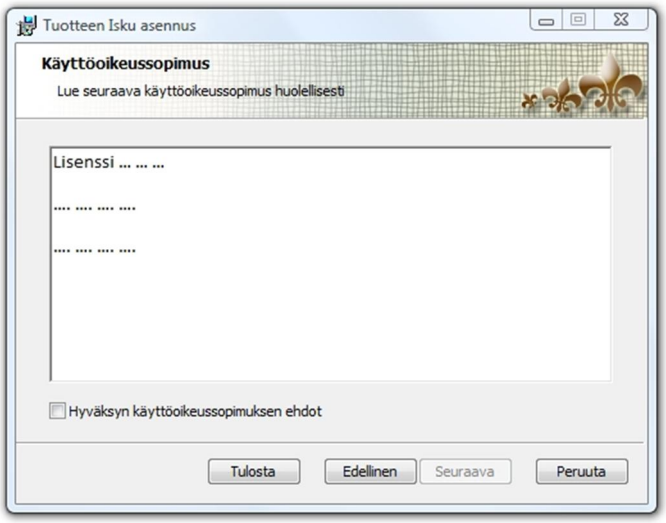
Jos Orca- ohjelmalla tutkimisen jälkeen asennuspaketissa on kaikki oikein, voidaan siirtyä toiseen testausmenetelmään eli sovelluksen asennukseen. Asennuksen testaus on hyvä suorittaa turvallisessa ja niin sanotussa puhtaassa ympäristössä. Esimerkkiympäristönä voisi toimia virtuaalikone. Ohjelmistokehittäjät käyttävät useimmin juuri virtuaalikoneita asennuspakettien testaukseen, koska asennuspaketeilla on vaikutusta käyttöjärjestelmään. Jokainen tiedosto siinä voisi tehdä muutoksen käyttöjärjestelmän asetuksiin, kuten rekisteriin, asetustiedostoihin, työpöytään jne. Testaaminen virtuaalikoneella oikean sijaan suojaa oikeaa käyttöjärjestelmää mahdollisista haitoista sekä virtuaalikoneen käyttöjärjestelmä on helpompi palauttaa käyttämällä varmuuskopioita. Lisäksi virtuaalikoneeseen on helpompi asentaa puhdas käyttöjärjestelmä eli ilman muita asennettuja sovelluksia. Usein virtuaalikonetta käytetään sovelluksen testaukseen eri käyttöjärjestelmissä: Windows XP:ssa, Windows Vista:ssa, Windows 7:ssa.

Kun testausympäristö on valittu, siirrytään testiasennukseen. Tehdään kaikki kuten asennettaessa tavallista ohjelmaa. Avataan Isku- asennuspakettitiedosto ja odotetaan asennuksen käyttöliittymän ensimmäisen dialogi-ikkunan ilmestymistä tietokoneruutuun (Kuva 9). Sen pitäisi sisältää taustakuvaa.



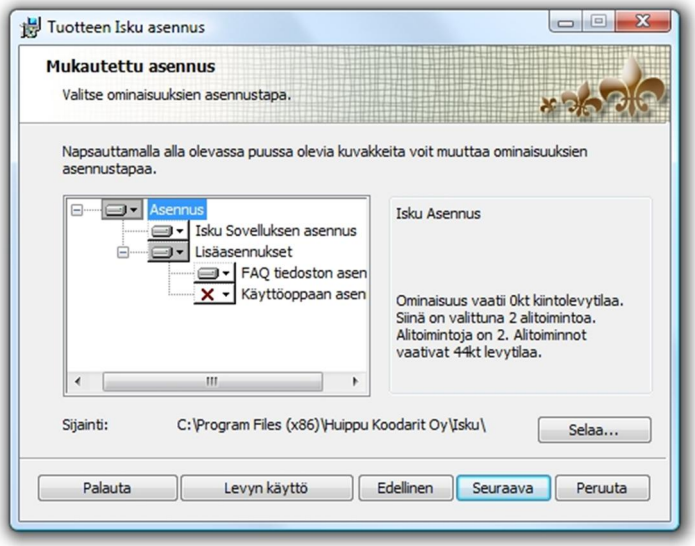
Kuva 9. Isku- asennuksen käyttöliittymän ensimmäinen dialogi-ikkuna.

Painetaan Seuraava- nappia ja jatketaan asennusta. Seuraavana pitäisi tulla toinen dialogi-ikkuna, joka sisältää ylhäällä toisen käyttöliittymää koristavan kuvan sekä lisenssitiedoston tekstin. Asennusta ei voisi jatkaa eteenpäin ilman lisenssiehtojen hyväksymistä (Kuva 10).



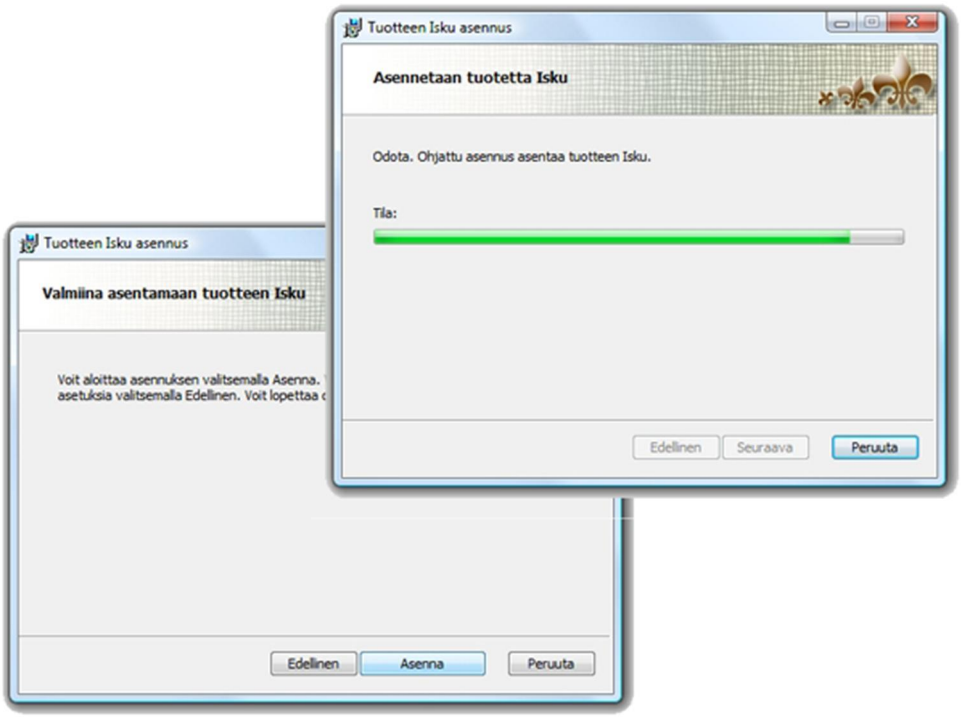
Kuva 10. Isku- asennuksen käyttöliittymän toinen dialogi-ikkuna.

Kolmantena, lisenssi dialogi-ikkunan jälkeen, tulee dialogi-ikkuna, jossa voidaan valita asennuksen tiedostojen sijainnin tulevan käyttäjän tietokoneella sekä mahdolliset asennettavat komponentit ja niiden asennusmuoto. Mahdolliset asennusmuodot sekä asennuksen tila eli asennetaanko tiedosto, on valitettavissa menuikkunasta, joka avautuu painamalla kuvaketta. Ylhäällä dialogi-ikkunassa pitäisi olla sama koristekuva kuten edellisessä dialogi-ikkunassa (Kuva 11).



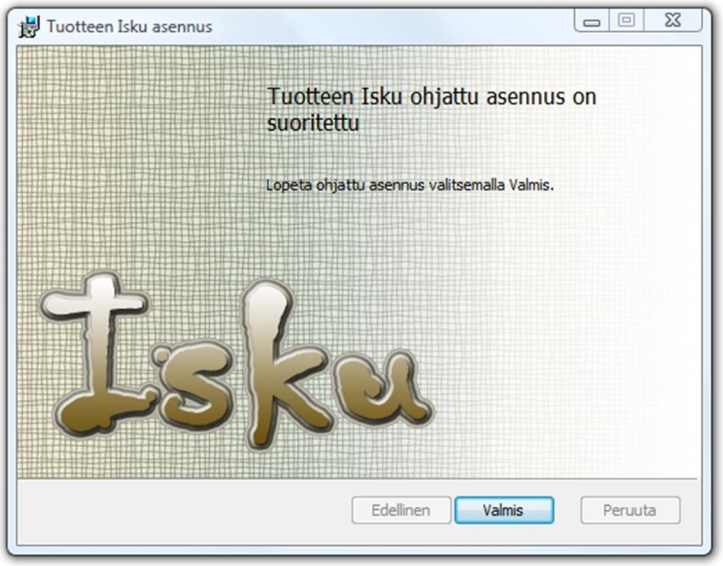
Kuva 11. Isku- asennuksen käyttöliittymän kolmas dialogi-ikkuna.

Seuraavana tulevat kaksi dialogi-ikkunaa, jotka varmistavat käyttäjiltä viimeisen kerran haluavatko he valittujen komponentien asennusta omalle tietokoneelle ja näyttävät asennusprosessin tilan (Kuva 12).Tutut yläreunan koristekuvat ovat mukana, kuten on suunniteltukin, että ne olisivat kaikkissa dialogi-ikkunnoissa, paitsi ensimmäisessä ja viimeisessä.



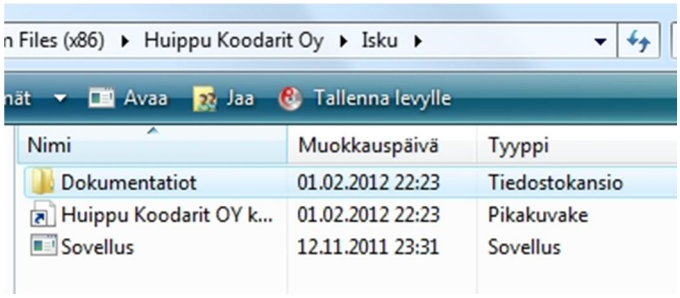
Kuva 12. Isku- asennuksen käyttöliittymän neljäs ja viides dialogi-ikkunat.

Kuudentena ja viimeisessä dialogi-ikkunassa taustalla on sama kuva kuin ensimmäisessä dialogi-ikkunassa, jonka eteen tulee teksti, joka kertoo onnistuneesta ohjelman asennuksesta. Käyttäjä ohjataan painamaan Valmis-nappia, jonka painamisen jälkeen tehtävänsä suorittanut asennuksen käyttöliittymä katoaa.



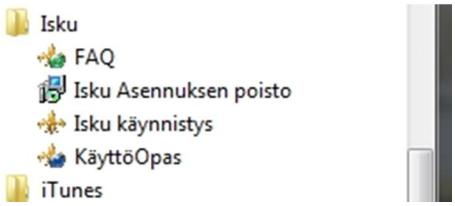
Kuva 13. Isku- asennuksen käyttöliittymän kuudes dialogi-ikkuna.

Suoritettujen toimintojen jälkeen käyttäjän tietokoneen työpöydälle pitäisi ilmestyä sovelluksen käynnistämiskuvake. Varsinainen sisältö pitäisi löytyä joko oletuksena Program Files- kansioon luoduista tekijän ja tuotteen nimisistä kansioista tai asennuksen aikana käyttäjän määrittelemästä osoitepolusta (Kuva 14). Lisäksi pikaisesti sisältöön pääse Käynnistä- valikkoon ilmestyneen Isku- kansion avulla. Jos kaikissa luetelluissa paikoissa löytyvät odotetut tiedostot, voidaan olettaa asennusohjelman tehneen kaikki sen tehtävät. Jää vain itse sovelluksen toimivuuden testaus, mutta se ei enää liity itse asennuspakettiin.



Kuva 14. Kaiken mahdollisen jaettavan sisällön oletus asennuspaikka käyttäjätietokoneella.

Testataan vielä lopuksi Isku- asennuspaketilla jaetun sisällön poistoa käyttäjän tietokoneelta. Sisällön poisto onnistuu ohjauspaneelista käyttäen ”Lisää tai poista”- sovellusta. Sen lisäksi kaikki voidaan poistaa pikaisesti Käynnistä- valikon Isku- kansion alta löytyvän ja kehittäjän suunnitelmien mukaan luodun poistokuvakkeen avulla (Kuva 15). Onnistuneen poistamisen jälkeen kaikki testauksen vaiheet on käyty läpi.



Kuva 15. Käyttäjän tietokoneen Käynnistä- valikkoon ilmestyvä Isku- tuotekansio sekä poistokuvake.

5.4 Arvio

Asennusohjelman esimerkin tavoitteena oli näyttää, miten käyttäen edellä opittuja perustietoja WiX- teknologiasta jokainen pystyisi luomaan oman asennuspaketin, joka ei toiminnoiltaan, eikä ulkonäöltään erottuisi markkinoilla jaettavista tuotteista.

Näiden tavoitteiden saavuttamiseksi valmistettiin etukäteen lisämateriaalit: kuvakkeet, taustakuvat jne. Opittuja WiX teknologian perustietoja sovellettiin esimerkissä lisäämällä niihin personalisoinnin piirteitä. Itse asennuspaketin tekoon kuluva aika oli noin puoli tuntia. Tuloksena syntyi tavoitteita vastaava asennusohjelman sisältävä asennuspaketti. Se soveltuisi käyttöön nykyisessä muodossaan pienyrityksille heidän tuotteiden jakelua varten.

6 Yhteenveto

Luettuaan tämän opinnäytetyön läpi lukijoilla tulisi olla riittävä yleinen tieto asennuspaketeista, jotka ajavat WiX- teknologiaa käyttäen luotua asennusohjelmaa ja sisältävät jaettavilla olevia tiedostoja, jotta he pystyisivät luomaan oman asennuspaketin. Lukijoita on tutustutettu Windows- tietokäyttöjärjestelmien asennusohjelmien luomisen tapoihin, Windows Installer- palveluun, sekä MSI- tiedostojen rakenteeseen, joka on purettu Orca- ohjelmalla. Kaikkien käytettyjen työkalujen kohdalla on selitetty niiden saantimahdollisuudet. Sovelluskehitysalustojen kohdalla on mainittu kaksi vaihtoehtoa, joita voi valita halujen ja saantimahdollisuuksien mukaan. Myös muut mahdollisuudet luoda asennusohjelma Windows- tietokonejärjestelmää varten, kuin WiX- työkalupaketin käyttö, ovat mainitut, selitetyt ja verratut keskenään tekstissä. Tekstissä teorian ja esimerkkiasennuksen kohdilla on käytetty runsaasti näyte-esimerkkejä koodipätkistä helpottaakseen lukijaa oppimaan tiedot myös visuaalisesti.

WiX- teknologia ei lähtökohtaisesti ole vaikeaa, mutta siinä on paljon nyansseja. Jotkut tyytyvät perustietojen oppimiseen, koska ne riittävät toimivan asennuksen luomiseen. Jotkut taas voivat jatkuvasti oppia uutta, koska teknologia on laaja ja aina kehityksen alla. Voidaan ennustaa, että WiX- tuotteena ei katoa markkinoilta lähitulevaisuudessa, joten sen oppiminen hyödyttäisi sekä aloittelijoita että sovellusten ja asennuspakettien valmistajia.

Lähteet

Kirjat:

Baker Bob. 2002.The Official InstallShield for Windows Installer Developer's Guide. Foster. M&T Books.

Ramirez Nick. 2010. WIX Developer's Guide to Windows Installer XML. Birmingham. Packt Publishing

Wilson Phil. 2004. The Definitive Guide to Windows Installer. New York. Apress.

Elektroniset lähteet:

Jordan Russell Software. 2011. Inno Setup. Viitattu 20.12.2011. http://www.jrsoftware.org/isinfo.php

Matt Ward, SharpDevelop Community. 2006. Feature Tour. Viitattu 29.1.2012. http://community.sharpdevelop.net/blogs/mattward/articles/FeatureTour.aspx

Microsoft Developer Networkin kirjasto. 2012. Windows Installer. Viitattu 24.1.2012. http://msdn.microsoft.com/en-us/library/cc185688%28VS.85%29.aspx

Microsoft Developer Networkin kirjasto. 2012. Locale IDs Assigned by Microsoft. Viitattu 16.1.2012. http://msdn.microsoft.com/en-us/goglobal/bb964664

Nullsoft Scriptable Install System, 2011. Etusivu. Viitattu 20.12.2011 http://nsis.sourceforge.net/Main_Page

Nullsoft Scriptable Install System. 2011. Users. Viitattu 20.12.2011. http://nsis.sourceforge.net/Users

SourceForge.net. 2006. Project of the Month, January 2006. Viitattu 20.12.2011. http://sourceforge.net/potm/potm-2006-01.php

Windows Installer XML (WiX) toolset Documentation, Manual (WiX 3.x). 2010. File Types. Viitattu 18.1.2012. http://wix.sourceforge.net/manual-wix3/files.htm

Windows Installer XML (WiX) toolset Documentation, Manual (WiX 3.x). 2010. How To: Create a Shortcut to a Webpage. Viitattu 11.2.2012. http://wix.sourceforge.net/manual-wix3/create_internet_shortcut.htm

Windows Installer XML (WiX) toolset Documentation, Manual (WiX 3.x). 2010. How To: Create a Shortcut on the Start Menu. Viitattu 10.2.2012. http://wix.sourceforge.net/manual-wix3/create_start_menu_shortcut.htm

Windows Installer XML (WiX) toolset Documentation, Manual (WiX 3.x). 2010. How To: Create an Uninstall Shortcut. Viitattu 10.2.2012. http://wix.sourceforge.net/manual-wix3/create_an_uninstall_shortcut.htm

Windows Installer XML (WiX) toolset Documentation, Manual (WiX 3.x). 2010. Customizing Built-in WixUI Dialog Sets. Viitattu 2.3.2012. http://wix.sourceforge.net/manual-wix3/WixUI_customizations.htm

Windows Installer XML (WiX) toolset Documentation, Manual (WiX 3.x). 2010. Feature Element. Viitattu 22.1.2012. http://wix.sourceforge.net/manual-wix2/wix_xsd_feature.htm

Liitteet

Liite 1. Isku asennuspaketin koodi. 39

Liite 1. Isku asennuspaketin koodi.

Toimiva asennuspaketin koodi, joka on luotu tehtäessä opinnäytetyön esimerkkiä.

```
<?xml version="1.0" encoding="UTF-8"?>
<Wix xmlns="http://schemas.microsoft.com/wix/2006/wi"
  xmlns:util="http://schemas.microsoft.com/wix/UtilExtension">
  <Product Id="17a07542-7e9e-40ea-a45a-dae1cbc04b34" Name="Isku" Language="1035"
Version="1.0.0.0" Manufacturer="Huippu Koodarit Oy" UpgradeCode="19327428-c3f5-
45dc-b9fc-aea1259a2141">
    <Package InstallerVersion="200" Manufacturer="Huippu Koodari Pekka" Com-
pressed="yes"/>
     <Media Id="1" Cabinet="media1.cab" EmbedCab="yes" />
    <!-- Asennushakemistot -->
    <Directory Id="TARGETDIR" Name="SourceDir">
      <!-- Hakemisto ProgramFiles\Huippu Koodarit Oy\Isku   -->
      <Directory Id="ProgramFilesFolder">
        <Directory Id="YrityksenKansio" Name="Huippu Koodarit Oy">
            <Directory Id="ASENNUSKANSIO" Name="Isku">
              <Directory Id="FAQKansio" Name="Dokumentatiot"/>
            </Directory>
            </Directory>
          </Directory>

      <!-- Alahakemisto Käynnistä\Ohjelmat\Isku -->
      <Directory Id="ProgramMenuFolder" Name="Programs">
        <Directory Id="ProgramMenuDir" Name="Isku"/>
      </Directory>
      <!-- Työpöytä  -->
    <Directory Id="DesktopFolder" Name="Desktop"/>
    </Directory>

        <!-- Käyttössä olevat ikonit -->
    <Icon Id="iskukuvake.ico" SourceFile="isku.ico"/>
    <Icon Id="opaskuvake.ico" SourceFile="iskuinfo.ico"/>
    <Icon Id="FAQkuvake.ico" SourceFile="iskufaq.ico"/>

    <!-- Varsinainen asennettava sisältö  -->
    <DirectoryRef Id="ASENNUSKANSIO">
      <Component Id="IskuSovellus" Guid="1603164F-1BC8-4350-A005-A82487610B24">
        <File Id="IskuTiedosto" Source="Sovellus.exe" DiskId="1" KeyPath="yes">
          <!-- Sovelluksen kuvakkeet käynnistämistä varteen -->
          <Shortcut Id="PikakaynnistysKuvake" Name="Isku käynnistys"
          WorkingDirectory="Isku" Directory="ProgramMenuDir" Advertise="yes"
          Icon="iskukuvake.ico"/>
          <Shortcut Id="TyopoytaKuvake" Name="Isku" WorkingDirectory="Isku" Di-
rectory="DesktopFolder" Advertise="yes"  Icon="iskukuvake.ico"/>
        </File>
      </Component>
      <!-- Hyperlinkki valmistajiin kotisivuihin -->
      <Component Id="Valmistajankotisivut" Guid="BD85BCFB-5AD7-49D6-A70F-
0098D2A0AC74">
        <util:InternetShortcut Id="Linkkikuvake" Name="Huippu Koodarit OY kotisi-
vut" Target="http://www.google.com/"/>
      </Component>
    </DirectoryRef>
    <!-- Dokumentatiot hakemiston sisältö -->
    <DirectoryRef Id="FAQKansio">
      <Component Id="KayttoOpas" Guid="389A9E01-89C8-41F3-970E-269560D6678B">
        <File Id="KayttoOpastiedosto" Name="käyttöopas.pdf" DiskId="1"
Source="käyttöopas.pdf" KeyPath="yes">
          <Shortcut Id="Opaskuvake" Name="KäyttöOpas" WorkingDirecto-
ry="FAQKansio" Directory="ProgramMenuDir" Advertise="yes" Icon="opaskuvake.ico"
/>
```

```
        </File>
      </Component>
      <Component Id="FAQ" Guid="AF60B531-22FC-42C3-90DA-2F7A07F2155A" Loca-
tion="either">
        <File Id="FAQtiedosto" Name="FAQ.txt" Source="FAQ.txt" DiskId="1" Key-
Path="yes" Compressed="no">
          <Shortcut Id="FAQtiedostokuvake" Name="FAQ" WorkingDirecto-
ry="FAQKansio" Directory="ProgramMenuDir" Advertise="yes" Icon="FAQkuvake.ico"/>
        </File>
      </Component>
    </DirectoryRef>
    <!-- Poisto kuvake -->
    <DirectoryRef Id="ProgramMenuDir">
      <Component Id="Poistaminen" Guid="4BE0BB91-5568-471C-AE0E-4DC582A6D640">
        <Shortcut Id="SovelluksenPoisto" Name="Isku Asennuksen poisto" Tar-
get="[Systen64Folder]msiexec.exe" Arguments="/x [ProductCode]"></Shortcut>
        <RemoveFolder Id="ProgramMenuDir" On="uninstall"/>
        <RegistryValue Root="HKCU" Key="Software\[Manufacturer]\[ProductName]"
Type="string" Value="1" KeyPath="yes"/>
        </Component>
    </DirectoryRef>

    <!-- Asennuksen vaihtoehdot -->
    <Feature Id="Asennus" Title="Asennus" Description="Isku Asennus" Dis-
play="expand" Level="1" ConfigurableDirectory="ASENNUSKANSIO" AllowAdvertise="no"
Absent="disallow" InstallDefault="local">
      <Feature Id="OletusAsennus" Title="Isku Sovelluksen asennus" Level="1" Al-
lowAdvertise="no" Absent="disallow" InstallDefault="local">
        <ComponentRef Id="IskuSovellus"/>
        <ComponentRef Id="Valmistajankotisivut"/>
        <ComponentRef Id="Poistaminen"/>

      </Feature>
      <!-- Vaihtoehtoiset asennukset -->
      <Feature Id="LisaAsennukset" Title="Lisäasennukset" Description="Asentaa
lisäasennukset käyttäjän valinnan mukaan"  Level="2" AllowAdvertise="no" Ab-
sent="disallow" InstallDefault="source">
        <Feature Id="FAQAsennus" Title="FAQ tiedoston asennus" Description="FAQ
tiedosto sisältä useimmin kysytyt kysymykset Isku sovelluksen liittyen" Level="2"
AllowAdvertise="no" Absent="disallow" InstallDefault="source">
          <ComponentRef Id="FAQ"/>
        </Feature>
        <Feature Id="OpasAsennus" Title="Käyttöoppaan asennus" Descripti-
on="Asentaa käyttöoppaan käyttäjän apuvälineksi" Level="2" AllowAdverti-
se="system" Absent="allow" InstallDefault="local">
        <ComponentRef Id="KayttoOpas"/>
      </Feature>
      </Feature>
    </Feature>
    <!-- Asennuksen käyttöliitymä -->
    <UI Id="IskunKayttoliityma">
      <UIRef Id="WixUI_FeatureTree"/>
      <UIRef Id="WixUI_ErrorProgressText"/>
      </UI>
    <!-- Kustomointi-->
    <WixVariable Id="WixUIBannerBmp" Value="isku5.bmp" />
    <WixVariable Id="WixUILicenseRtf" Value="lisenssi.rtf" />
    <WixVariable Id="WixUIDialogBmp" Value="isku6.bmp"/>
  </Product>
</Wix>
```