OHJELMISTON ASENNUSTYÖKALUT

LAHDEN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma
Ohjelmistotekniikka
Opinnäytetyö
Kevät 2009
Petteri Tamminen

Lahden ammattikorkeakoulu
Tietotekniikan koulutusohjelma

TAMMINEN, PETTERI: Ohjelmiston asennustyökalut

Ohjelmistotekniikan opinnäytetyö, 70 sivua, 12 liitesivua

Kevät 2009

TIIVISTELMÄ

Erityisesti Windows-käyttöjärjestelmissä ohjelmistojen käyttöönotossa on näkyvästi esillä asennusohjelma, jonka tarkoituksena on integroida ohjelmisto osaksi järjestelmää ohjelmiston käyttämiseksi sekä mahdollistaa ohjelmiston korjaaminen ja poistaminen järjestelmästä. Vaikka asennusohjelma ei ole osa asennettavaa ohjelmistoa, sillä on vaikutusta käyttäjille muodostuvaan ensivaikutelmaan ohjelmistosta. Tietyillä käyttäjäryhmillä, kuten järjestelmänvalvojilla, on itse asennusohjelman toimintaan liittyviä odotuksia.

Linuxissa ja Mac OS:ssa ohjelmistojen käyttöönotto poikkeaa useimmiten Windowsin vastaavasta, mutta myös näissä järjestelmissä on ohjelmistojen asentamiseen tarkoitetut sovelluksensa. Windowsissa asennusohjelmistojen teknologioita on useita, joista Microsoftin oma Windows Installer -teknologia on yksi käytetyimmistä. Sen käyttö on myös vaatimuksena haettaessa ohjelmistolle ”Windows Logo” -sertifikaattia.

Windows Installer -teknologia poikkeaa huomattavasti muista asennusteknologioista. Perinteiset teknologiat ovat proseduraalisia perustuen erilaisiin asennusskripteihin, Windows Installer puolestaan on tietokantapohjainen ja toimintamalliltaan deklaratiivinen, jossa asennuksen alussa määritellään järjestelmän tila asennusoperaatioiden jälkeen. Mikäli tilaa ei saavuteta, järjestelmä palautetaan lähtötilaan. Microsoftin omana teknologiana Windows Installer on liitetty tiukasti osaksi käyttöjärjestelmää, ja se sisältää lukuisia ohjelmistotuotteisiin ja ohjelmakomponentteihin liittyviä ominaisuuksia.

Nokian PC Suite -ohjelmiston asennus on toteutettu Windows Installerilla. Useista asennuspaketeista koostuvana ohjelmistona sen asentamiseksi kehitettiin Nokia Installer -ohjelma, jonka avulla eri asennukset saadaan asennettua samasta sovelluksesta, ja käyttäjälle välittyy vaikutelma yhdestä asennettavasta ohjelmistotuotteesta. Myöhemmin Nokia Installerin haluttiin soveltuvan muidenkin ohjelmistojen asentamiseen, joten sitä päivitettiin ja sen käytön tueksi suunniteltiin konfigurointia helpottavia aputyökaluja.

Tämän työn tavoitteena oli toteuttaa helppokäyttöinen, graafinen työkalusovellus Nokia Installerin asennuspakettien sisällön muokkaamiseen esimerkiksi matkapuhelinten mukana toimitettavan ohjelmistosisällön räätälöimiseksi. Työ toteutettiin C# -sovelluksena ja tuloksena oli toimiva työkalusovellus, jota ei suunnitelmamuutosten takia kuitenkaan otettu käyttöön.

Avainsanat: asennusohjelma, installer, msi, Windows Installer

Lahti University of Applied Sciences
Faculty of Technology

TAMMINEN, PETTERI: Software Installation Tools

Bachelor's Thesis in Software Engineering, 70 pages, 12 appendices

Spring 2009

ABSTRACT

Software products, especially concerning Windows operating systems, mostly include an installation program, whose purpose is to integrate the software product with the operating system in order for the user to be able to use the software. The installation program is also used to correct errors and to remove the software from the system. Even though the installation program is not part of the software being installed by it, it still has an influence on the user's first impression of the software product. Also, certain user groups, such as IT administrators, have expectations concerning the installation program itself.

In Linux and Mac operating systems, software installations are usually different compared to Windows, but these systems also have applications for installing software. In the Windows environment, several installation technologies exist, among which the "Windows Installer" developed by Microsoft is one the most widely used. It is also required to be used in order to obtain the "Windows Logo Program" certificate.

The Windows Installer technology differs radically from traditional installation technologies. Traditional technologies are procedural, based on installation scripts, while Windows Installer is database oriented and uses a declarative approach, which defines the state of the installation action in the beginning of the installation process. Unless the defined state can be reached, the system is reverted back to the original state. Since the Windows Installer technology was developed by Microsoft, it is tightly integrated with the underlying operating system, providing numerous features related to software installation operations.

Nokia PC Suite software installation is implemented using Windows Installer. Nokia PC Suite includes multiple installation packages and in order to install it as a single product from the user's perspective, an application called Nokia Installer was developed. Later on, Nokia Installer was updated to be suitable for installing other software as well and certain helper tools were designed for it.

In this thesis the purpose was to implement an easy-to-use application with graphical user interface for making custom configurations for Nokia Installer. For instance, the tool could be used for creating tailored software packages for different mobile phone models. The tool was implemented as a C# application. The result of the project was a working application. However, due to changes in the plans the tool was not taken into use.

Key words: installation program, installer, msi, Windows Installer

SISÄLLYS

TERMIT JA LYHENTEET

API	Application Programming Interface, ohjelmointirajapinta.
Bootstrapper	Asennusohjelman käynnistävä sovellus.
CAB	Cabinet-tiedostotyyppi. Windowsin tiedostopakkauksen oletus-tyyppi.
CCD	Connectivity Cable Driver. Laiteajurit Nokian matkapuhelimen yhdistämiseksi PC-tietokoneeseen kaapelilla.
COM	Component Object Model. Microsoftin kehittämä Windows -objektien binääristandardi.
FAQ	Frequently Asked Questions – usein kysyttyjä kysymyksiä.
GUID	Globally Unique Identifier. Sarjasta heksalukuja koostuva, globaalisti yksilöivä tunniste.
ICE	Internal Consistency Evaluator. Windows Installerin mukautettu toiminto, joka tarkistaa msi-tietokannan oikeellisuutta.
Installeripaketti	Nokia Installerilla asennettavaksi tarkoitettu kokoelma Windows Installer asennuksia.
Jakelupaketti	Ohjelmisto, jonka jakaminen kolmannen osapuolen toimesta on sallittua ohjelmiston lisenssin puitteissa.
LZMA	Lempel-Ziv-Markov -algoritmi.
MFC	Microsoft Foundation Classes. C++ -luokkakirjasto.
MSI	Microsoft Software Installation. Windows Installer -asennusteknologia.
PCCS	PC Connectivity Solution. PC-tietokoneen ja Nokian matkapuhelimen välisestä kommunikoinnista vastaa ohjelmisto.
SDK	Software Development Kit.
SID	Security Identifier. Windows-käyttöjärjestelmissä käyttäjän yksilöivä suojaustunniste.
SP	Service Pack. Ohjelmiston päivityspaketti.
USB	Universal Serial Bus. Sarjaväyläarkkitehtuuri oheislaitteiden liittämiseksi tietokoneeseen.
Active Directory	Windowsin käyttäjätietokanta ja hakemistopalvelu resurssien jakamiseksi verkon käyttäjille.

1 JOHDANTO

Ketterien ohjelmistokehitysmenetelmien ja jatkuvan integroinnin yleistyminen on tehostanut ohjelmiston automatisoitua valmistusprosessia lähdekoodista asennuspakettiin. Ohjelmiston päivittäminen uudella asennuspaketilla on monesti kustannustehokkaampaa kuin erillisten päivityspakettien tekeminen. Ohjelmistotuotannossa onkin ollut havaittavissa muutosta suuntaan, jossa ohjelmisto pilkotaan erikseen asennettaviin ja siten myös päivitettäviin osakokonaisuuksiin. Muutos on vaatinut asennusohjelmien mukauttamista useiden asennuspakettien asentamiseksi näennäisesti yhtenä tuotteena ja luonut myös tarvetta asennuksien konfigurointiin tarkoitetuille sovelluksille.

Tässä opinnäytetyössä toteutetaan konfiguraatiotyökalu Nokia Installerille. Työkalun on tarkoituksena olla helppokäyttöinen soveltuen muidenkin kuin vain insinöörien käytettäväksi. Lisäksi perehdytään ohjelmistojen asennuksiin Windows -järjestelmissä, muita vallitsevia järjestelmiä tarkastellaan pintapuolisesti selkeyttämään järjestelmien merkittävimpiä eroja.

Nokia Installer kehitettiin ratkaisuksi useista Windows Installer -asennuspaketeista koostuvan ohjelmistokokonaisuuden hallitsemiseksi. Se suunniteltiin alun perin Nokia PC Suitea varten, mutta sitä laajennettiin myöhemmin soveltuvaksi muidenkin Nokian ohjelmistojen asentamiseen ja sen konfigurointia haluttiin helpottaa ulkoisilla työkalusovelluksilla. Työ tehdään Nokian Services & Software yksikössä Tampereella.

Nokia on langattoman viestinnän tekniikkaa ja palveluja tuottava yritys, jolla on tutkimus- ja kehittämistoimintaa kymmenessä maassa sekä tuotanto-, myynti- ja markkinointitoimintoja maailmanlaajuisesti. Päätuotteita ovat matkaviestimet ja kuluttajille suunnatut Internet-palvelut sekä yritysratkaisut. (Nokia lyhyesti 2008.)

2 OHJELMISTOTUOTTEEN ASENNUS

2.1 Yleistä

Integrointi ja *käyttöönotto* ovat osa ohjelmistotuotteen kehitystä. Tässä yhteydessä integroinnilla tarkoitetaan vaihetta, jossa lähdekoodeista käännetyt ohjelmakomponentit sekä muut ohjelmiston tarvitsemat tiedostot kerätään yhdeksi kokonaisuudeksi ja asetetaan toimimaan samassa järjestelmässä. Ohjelmistosta riippuen integrointi voi vaatia useita erilaisia toimenpiteitä kohdejärjestelmässä, kuten tiedostojen kopiointia määrättyyn hakemistorakenteeseen, tietokantojen luontia, ajonaikaisten kirjastojen tai laiteohjaimien asentamista, ympäristömuuttujien asettamista, asetustiedostojen tai järjestelmärekisterin muokkausta. Integrointiin tarvittavat toimenpiteet muodostavat merkittävän osan ohjelmiston käyttöönoton vaatimuksista ja ne pyritään automatisoimaan, jotta käyttöönotto olisi helppoa. Käyttöönottoon liittyy monesti ohjelmiston asennusohjelma tai pakettienhallintaohjelma, jonka tehtävä on integroida ohjelmisto käyttöjärjestelmään.

2.2 Asennusohjelman tarkoitus

Asennusohjelmalla on tietyt perustoiminnot riippumatta käytössä olevasta käyttöjärjestelmästä. Sen tarkoituksena on:

- Opastaa käyttäjää ohjelmiston asennuksessa, tavallisesti velhotyylisen, graafisen käyttöliittymän avulla.
- Varmistaa ennen asentamista, että kohdejärjestelmä täyttää ohjelmistolle asetetut vaatimukset.
- Kopioida ohjelmiston tiedostot asennuspaketista kohdejärjestelmään.
- Luoda ohjelmistolle suoritusympäristö tekemällä tarpeelliset muutokset kohdejärjestelmään.
- Tarjota käyttäjille tapa käynnistää ohjelmisto helposti.
- Tarjota käyttäjille tapa poistaa ohjelmisto järjestelmästä. (Baker 2001, 5.)

2.3 Asennuksen käyttäjät

2.3.1 Käyttäjäryhmät

Peruskäyttäjäksi katsotaan henkilö, jolla on perustiedot tietokoneen käytöstä. Hän tietää ohjelmistojen asentamisen käsitteen, mutta ei omaa laajempaa tietämystä aiheesta. Edistyneet käyttäjät ovat perehtyneet tietokoneen laitteistoon, käyttöjärjestelmään ja ohjelmiin peruskäyttäjiä syvällisemmin. Peruskäyttäjät ja edistyneet käyttäjät muodostavat ohjelmiston loppukäyttäjien ryhmän. Heidän lisäkseen asennuksen kanssa tekemisissä ovat tietoverkon järjestelmävalvojat sekä ohjelmistojen asennuksia tekevät asennuskehittäjät. Kullakin ryhmällä on toisistaan poikkeavia odotuksia ohjelmiston asennukselta.

2.3.2 Loppukäyttäjä

Asennus voi olla loppukäyttäjän ensimmäinen kosketus asennettavaan ohjelmistoon, jolloin sillä on jo vaikutusta ohjelmistosta muodostuvaan ensivaikutelmaan. Helppokäyttöisen ja luotettavan asennuksen on todettu parantavan asiakastyytyväisyyttä sekä vähentävän ohjelmiston tukipalveluiden työtä (Baker 2002, 5).

Loppukäyttäjien odotuksia ohjelmiston asennukselta:

- Asentaminen ja poistaminen on yksinkertaista. Asennus ei esitä kysymyksiä, joihin käyttäjä ei tiedä vastauksia.
- Asennus osaa päivittää aiemmin asennetun ohjelmistoversion.
- Asennus osaa korjata vioittuneen ohjelmiston.
- Asennuksen aikana on mahdollista valita, mitä osia ohjelmistosta asennetaan. (Baker 2002, 4.)

Käyttäjien kannalta ideaalisessa tapauksessa ohjelmiston asennus ei vaatisi käyttäjän vuorovaikutusta, toisin sanoen ohjelmisto asentaisi itse itsensä (Baker 2001, 4). Käytännössä tällaista tilannetta on vaikea saavuttaa tietoturvallisesti muissa kuin suljetuissa ympäristöissä.

2.3.3 Järjestelmävalvoja

Ohjelmistojen automatisoitu asennus yrityksen tai organisaation tietoverkossa vähentää työasemien ylläpitokustannuksia, joten tietoverkon järjestelmävalvojan on kyettävä asentamaan ohjelmistoja käymättä henkilökohtaisesti jokaisella koneella. Järjestelmävalvojalla on seuraavia odotuksia ohjelmistojen asennuksilta:

- Asennus voidaan suorittaa keskitetystä paikasta kaikkiin verkon työasemiin.
- Asennus on muokattavissa tai konfiguroitavissa siten, että vain halutut ominaisuudet ohjelmistosta asennetaan.
- Asennus voidaan suorittaa ilman käyttäjän osallistumista asennukseen tai, että käyttäjä on edes tietoinen meneillään olevasta asennuksesta.
- Asennuksen onnistuminen, epäonnistuminen ja virheet asennuksessa ovat nähtävissä keskitetystä paikasta. (Baker 2002, 5.)

2.3.4 Asennuskehittäjä

Asennuksen tekemistä ei aiemmin otettu huomioon osana ohjelmiston kehitysprosessia ja se jätettiin usein viimeiseksi tehtäväksi juuri ennen ohjelmiston julkaisua. Asennukset perustuivat erilaisiin skripteihin ja olivat niukasti tai ei lainkaan dokumentoituja. Niiden ylläpito muuttui ajan mittaan yhä työläämmäksi asennuksen siirtyessä henkilöltä toiselle. Erityisen haasteellisia olivat suuret ohjelmistoprojektit, joissa ohjelmiston kehittäminen oli hajautettu maantieteellisesti eri paikkoihin. Ohjelmiston komponentit toimitettiin asennuskehittäjälle, jonka tehtävä oli koota komponenteista toimiva ohjelmisto ja paketoida se asennuspakettiin. Asennustyökaluja kehittivät useat eri valmistajat eikä asennuksille ollut vakiintuneita käytäntöjä, joten eri työkaluilla tehdyt asennukset olivat käyttöliittymiltään vaihtelevia sekä riippuvaisia työkaluista, joilla ne oli tehty. Asennuskehittäjän kannalta ideaalista olisi, jos asennukset olisivat toiminnoiltaan yhdenmukaisia käytetyistä työkaluista riippumatta. (Baker 2001, 5; Baker 2002, 6.)

Nykyisin asennuskehittäjien työ on muuttunut käyttöjärjestelmiin integroitujen asennusohjelmien (Windows Installer, Mac Installer), kehittyneempien asennustyökalujen ja ohjelmistokehityksen uusien menetelmien (ketterät menetelmät, jatkuva integrointi) myötä suoraviivaisemmaksi ja itse asennuksia on myös kehitetty aiempaa käyttäjäystävällisemmiksi.

2.4 Asennukset eri käyttöjärjestelmissä

2.4.1 Yleistetty tapaus

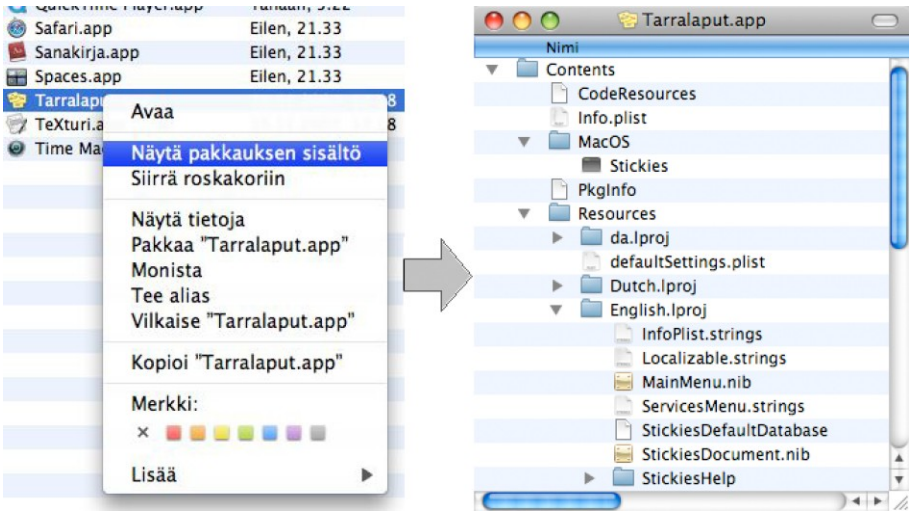
Windows, Mac OS ja Linux ovat tätä kirjoitettaessa markkinoita hallitsevat käyttöjärjestelmät (Operating System Market Share 2008). Järjestelmät eroavat toisistaan, joten ohjelmistojen käyttöönotto on erilaista kussakin järjestelmässä. Jos ohjelmistolle on käyttöönottoa varten tehty asennusohjelma, se voisi olla käyttöliittymältään yhdenmukainen eri käyttöjärjestelmissä, mutta sen sisäinen toiminta on aina käyttöjärjestelmän vaatimusten mukainen. Yksinkertaisimmillaan ohjelman käyttöönottoon kussakin käyttöjärjestelmässä riittää ohjelmatiedostojen kopiointi tietokoneen kiintolevylle ja ohjelman käynnistäminen.

2.4.2 Mac OS X

Applen Mac OS X -käyttöjärjestelmässä hakemistojen ja tiedostojen hallintaan on kehitetty pakkaustekniikka *(bundle mechanism)*, jossa tiedostojärjestelmän hakemisto sisältöineen näkyy OS X:n tiedostohallintaohjelma *Finderissa* yhtenä tiedostona. Finder tunnistaa pakatun hakemiston *ohjelmapakkaukseksi* ja käynnistää sen sisältämän ohjelman. Pakkaustekniikka otettiin käyttöön Mac OS 9:ssä yksinkertaistamaan ohjelmahakemiston sisäistä rakennetta. OS 9:n ohjelmapakkaus sisälsi ohjelmatiedoston, tarvittavat kirjastot ja resurssitiedostot. (Bundle Programming Guide 2005.)

OS X:ssä pakkaustekniikkaa on kehitetty edelleen ja se sisältää lukuisia uudistuksia, mm. tuen useille kielille ja eri Mac OS versioille. Uusien ominaisuuksien an-

siosta samaan pakkaukseen voidaan sisällyttää ohjelman OS 9 ja OS X -versiot sekä lokalisoidut kieliresurssit. Pakkauksen sisältö kuvataan *info.plist* -nimisessä XML-tiedostossa. Kuviossa 1 on esitetty osa tarralappuohjelman pakkausta. Ohjelma näkyy Finderissa yhtenä tiedostona, jolle avautuvasta ponnahdusvalikosta saadaan haluttaessa näkyviin ohjelmapakkauksen sisältö. (Bundle Programming Guide 2005.)



KUVIO 1. Mac OS X -ohjelmapakkaus

Ohjelmien käyttöönotto ohjelmapakkauksina on Applen dokumentaatiossa mainittu *manuaalisena asennuksena* (*manual install, drag and drop install)*. Se on tehty erittäin helpoksi, käyttäjä vain tallentaa pakkauksen tietokoneelleen ja käynnistää ohjelman kaksoisnapsauttamalla ohjelmapakkausta. Ohjelma poistetaan siirtämällä ohjelmapakkaus roskakoriin. Ensimmäisellä käynnistyskerralla ohjelma luo tarvittavat asetustiedostot käyttäjäkohtaiseen kirjastohakemistoon. Näitä tiedostoja ei poisteta automaattisesti, kun pakkaus poistetaan koneelta. Jos ohjelma otetaan myöhemmin uudelleen käyttöön, käyttäjän aiemmat asetukset ovat edelleen käytettävissä. (Software Delivery Guide 2006.)

Manuaalinen asennus oli ennen OS X -käyttöjärjestelmän versiota 10.5 (Leopard) suositeltu ohjelmien asentamistapa, mutta Leopardissa ja sitä uudemmissa käyttöjärjestelmän versioissa suositeltu tapa on *hallittu asennus* (*managed install*) (PackageMaker User Guide 2007). Siinä ohjelmisto pakataan asennuspakettiin,

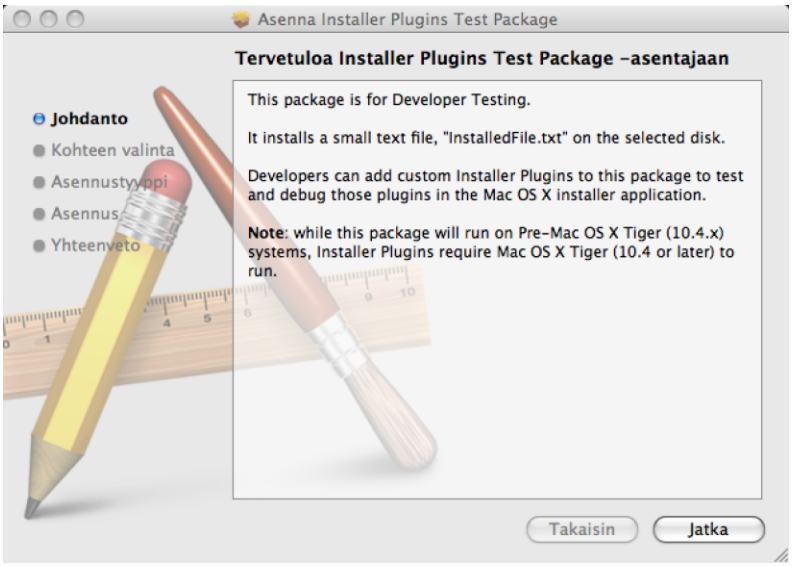
joka sisältää itse ohjelmiston lisäksi ohjeet *Asentajalle (Installer)* siitä, miten ohjelmisto tulee asentaa ja millaisia valintoja käyttäjälle tarjotaan. Hallittu asennus tarjoaa mm:

- Automatisoidun asennuksen useista ohjelmakomponenteista koostuville ohjelmistoille.
- Mahdollisuuden päivittää yksittäisiä ohjelmakomponentteja.
- Mahdollisuuden valita mitä osia ohjelmistosta asennetaan ja minne ne asennetaan.

Toisin kuin manuaalisessa asennuksessa, jossa kaikki ohjelman tiedostot sijaitsivat ohjelmapakkauksen sisällä, hallittu asennus voi asentaa tiedostoja myös järjestelmän muihin osiin. Tällöin asentaja vaatii käyttäjältä järjestelmänvalvojan oikeudet.

Asentaja on OS X:n ohjelma ohjelmistojen asentamiseen. Se ylläpitää listaa asennetuista ohjelmakomponenteista ja siten mahdollistaa niiden päivittämisen (PackageMaker User Guide 2007). Asentaja tarjoaa velhotyyppisen, graafisen käyttöliittymän, jonka ulkoasua ja vaiheita voi rajoitetusti muuttaa asennuspakettia tehtäessä. Lisäksi on mahdollista määrittää ennen asennusta tai asennuksen jälkeen suoritettavia toimintoja, esimerkiksi järjestelmävaatimusten tarkistaminen ennen asennusta tai asennetun ohjelmiston näyttämisen Finderissa asennuksen päätyttyä. Asennuspakettien tekoon käytätettävä *PackageMaker* kuuluu Mac OS:n mukana toimitettaviin kehitystyökaluihin. (PackageMaker User Guide 2007.)

Asentajan suomenkielinen käyttöliittymä on esitetty kuviossa 2. Ikkunan vasemmassa laidassa oleva lista näyttää mitä vaiheita asennuksessa on ja mikä vaihe on meneillään. Oikealla puolella on varattu tilaa vaiheen kuvaukselle. Vaiheiden välillä liikutaan ”Takaisin”- ja ”Jatka”-painikkeilla. Näiden lisäksi ikkunassa voi olla painikkeita muille toiminnoille.



KUVIO 2. Mac OS X asentaja

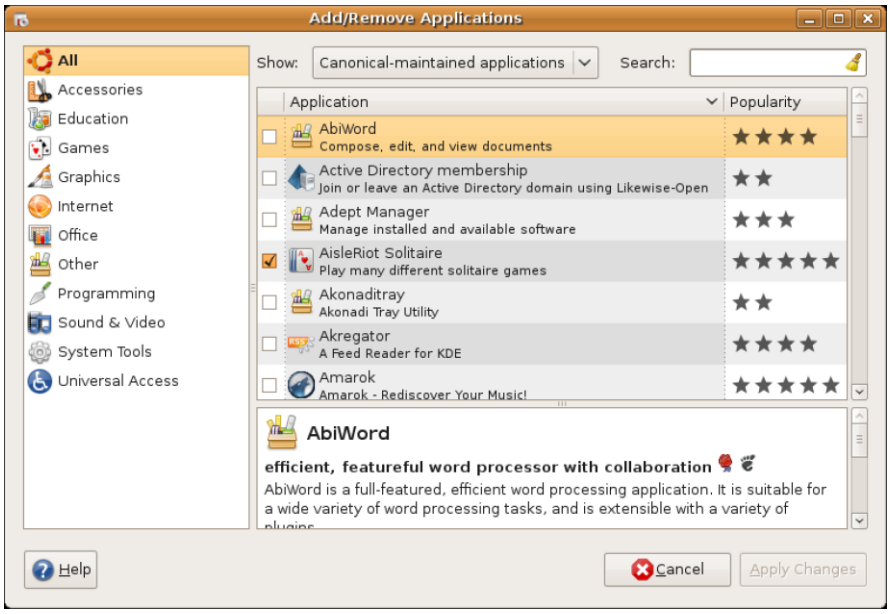
Asentajan rajoituksista mainittakoon, että se toimii vain Mac OS X -versioissa, eikä se tue asennetun ohjelmiston poistamista tietokoneelta (Software Delivery Guide 2006). Ohjelmiston poistamiseen tarvitaan siten erillinen apuohjelma, joka tavallisesti asennetaan ohjelmiston mukana.

2.4.3 Linux

Linux jakeluversioista vertailukohteeksi valittiin Ubuntu Linuxin versio 8.10. Se on Debian Linuxiin pohjautuva GNU/Linux-jakelupaketti, joka pyrkii tarjoamaan helposti käyttöönotettavan ja käytettävän, säännöllisesti päivittyvän Linux -käyttöjärjestelmän (Ubuntu–Wikipedia 2008).

Ohjelmistojen asennus, päivitys ja poisto tapahtuu pakettienhallintaohjelmalla, joka huolehtii Ubuntun *ohjelmapakettien* ja niiden riippuvuuksien asentamisesta Internetissä sijaitsevista ohjelmalähteistä (*repositories*) tietokoneeseen. Pakettienhallinta myös ilmoittaa käyttäjälle, mikäli asennettuihin ohjelmistoihin on saatavilla päivityksiä. Pakettienhallinnan käyttämiseen on kolme eri sovellusta, joista vain yhtä voidaan käyttää kerrallaan:

- *Add/Remove Applications*, joka on perusohjelma ohjelmistojen asentamiseen (kuvio 3).



KUVIO 3. Add/Remove Programs -ohjelma

- *Synaptic*, jossa on enemmän ominaisuuksia, kuten mahdollisuus asentaa palvelinohjelmistoja sekä asennettavan ohjelmistoversion valinta (kuvio 4).

KUVIO 4. Synaptic

- *APT (Advanced Packaging Tool)*, joka on komentorivipohjainen pakettienhallintasovellus.

Jos ohjelmistoa ei ole saatavissa ohjelmalähteistä, eikä siten ole asennettavissa pakettienhallinnan kautta, se voidaan ladata Internetistä ja asentaa yksittäisenä pakettina konsolin kautta. Asennettu ohjelmisto ei tällöin kuulu pakettienhallinnan piiriin, ja käyttäjän on itse huolehdittava sen päivittämisestä ja poistamisesta. (Ubuntu Documentation 2008.)

Itse ohjelmapaketti on .deb -päätteinen pakattu tiedosto, joka sisältää tiedostot *data.tar.gz* ja *control.tar.gz*. Data.tar.gz sisältää ohjelmiston suoritustiedostot ja dokumentaation, control.tar.gz puolestaan pakettienhallinnan tarvitsemat tiedot paketin versiosta, riippuvuuksista, ja siitä miten paketti asennetaan ja poistetaan. (Debian Administration 2006.)

2.4.4 Windows

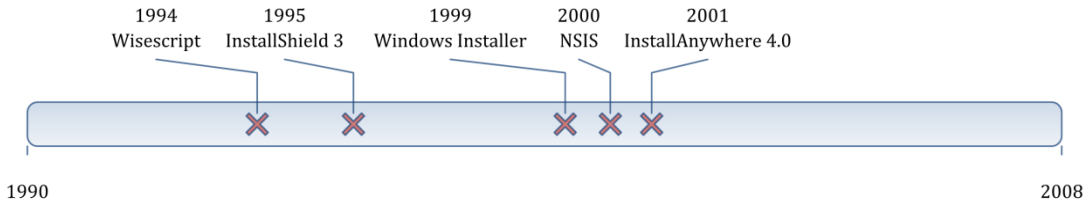
Windowsissa ei ole Mac OS:n tai Linuxin kaltaista pakettienhallintaa, vaan tavallisesti ohjelmistot asennetaan ja poistetaan yksitellen niiden omilla asennusohjelmilla. Useita ohjelmistoja sisältävissä kokonaisuuksissa saatetaan käyttää asentajasovellusta, jolla välillisesti hallinnoidaan ohjelmistoon kuuluvien tuotteiden asennuksia yhdestä sovelluksesta käsin. Asennuksia Windowsissa tarkastellaan tarkemmin kappaleessa 3.

Windowsin ohjauspaneelista löytyvä ”Lisää tai poista ohjelmia” -sovellus toimii keskitettynä paikkana asennettujen ohjelmistojen tarkastelemiseksi ja poistamiseksi. Uusien ohjelmien lisääminen sen kautta on mahdollista, jos käytössä on Active directory -hakemistopalvelu, jolloin järjestelmävalvoja voi julkaista ohjelmistoja palveluun liitettyjen käyttäjien asennettavaksi.

3 ASENNUKSET WINDOWS-JÄRJESTELMISSÄ

3.1 Vallitsevat tekniikat

Windowsissa asennusten tekoon löytyy erilaisia työkaluohjelmistoja, joista useimmat ovat kaupallisia, mutta myös avoimeen lähdekoodiin perustuvia ohjelmistoja on saatavilla. Kaupalliset ohjelmistot tukevat yleensä sekä Windows Installeria, että valmistajan omaa teknologiaa, mikäli sellainen on olemassa. Avoimen lähdekoodin ohjelmistot ovat erikoistuneet oman teknologiansa kehittämiseen. Kuvion 5 aikajanassa esiintyvät asennusteknologiat on kehitetty pääosin 1990-luvun puolella. Niitä kehitetään edelleen vastaamaan nykyisiä tarpeita ja vaatimuksia.



KUVIO 5. Asennusohjelmistoja aikajärjestyksessä

Wisen historia ulottuu kuvion 5 mukaisesti 1990-luvun alkupuolelle. Silloin käyttöönotettua *WiseScript*-tekniikkaa käytetään ja kehitetään edelleen ohjelmiston tukiessa myös Windows Installeria. Työkaluohjelmisto tarjoaa WiseScript -asennusten tekoon hiiriavusteisen tavan määrittää toimintoja asennusskriptiin, jonka ohjelmisto esittää englantia muistuttavassa, luettavassa muodossa. Valmis skripti käännetään Wisen kääntäjällä suoritettavaksi asennusohjelmaksi, josta asennusmoottorin koko on n.100 kilotavua. (WiseScript editor reference 2006.)

InstallShieldillä on Wisen tavoin pitkä historia 1990-luvun alkupuolelta. Sille kehitetty skriptikieli, *InstallScript*, on C-kieltä muistuttava, käännettävä skriptikieli, joka on vuosien saatossa laajentunut varsin kattavaksi kokoelmaksi valmiita funktioita. InstallShield tukee myös Windows Installeria sekä yhdistelmäasennus-

ta, jossa käytetään molempia teknologioita, sekä Windows Installeria että InstallScriptiä. (Baker 2002, 18–23)

Windows Installer on Microsoftin kehittämä asennusteknologia. Se liitettiin osaksi käyttöjärjestelmää Windows 2000 -versiosta lähtien tarkoituksena tarjota standardi tapa asentaa, ylläpitää ja poistaa ohjelmistoja. Skriptejä käyttävistä asennusohjelmista poiketen Windows Installer on tietokantapohjainen ja sen toimintamalli on deklaratiivinen: asennuksen yksittäisten vaiheiden ja operaatioiden määrittelyn sijaan määritellään tila, jossa järjestelmän tulee olla asennusoperaatioiden jälkeen (Wix tutorial 2004–2008). Mikäli tilaa ei saavuteta, järjestelmä palautetaan takaisin lähtötilaan. Epäonnistunut asennus ei siten jätä järjestelmää tuntemattomaan tilaan, jolloin muut ohjelmistot tai itse käyttöjärjestelmä vioittuu.

Windows Installer on asiakas–palvelin-tyyppinen ohjelma, jossa palvelin toimii Windowsin järjestelmäpalveluna ja asiakkaita ovat suoritettavat asennukset. Asennuspaketti, msi-tiedosto, on relaatiotietokanta sisältäen kaiken Windows Installerin tarvitseman informaation ohjelmiston asentamisesta ja poistamisesta.

Windows Installerin asennusarkkitehtuuri on avoin ja julkisesti dokumentoitu. Microsoft tarjoaa ohjelmointirajapinnat sekä pääosin komentorivipohjaiset kehitystyökalut, joita kolmansien osapuolten asennustyökaluohjelmistot käyttävät.

Windows Logo -sertifikaatin saamiseksi Microsoft edellyttää ohjelmiston käyttävän asennuksessaan Windows Installeria (Certified for Windows Vista Requirements 2006). Tämä on todennäköisesti vaikuttanut siihen, miksi uusia asennusteknologioita ei juuri ole esitelty Windows Installerin jälkeen.

NullSoft Scriptable Install System (NSIS) on avoimen lähdekoodin ohjelmisto, jonka suunnittelussa on kiinnitetty erityisesti huomiota asennuspakettien pieneen kokoon. Asennusmoottorin tuoma koon lisäys asennusohjelmaan on vain 34 kilotavua. NSIS tukee tehokasta LZMA-pakkausalgoritmia purkamiseen tarvittavan toiminnallisuuden sisältyessä em. 34 kilotavuun. (NSIS Users Manual 2008.)

NSIS käyttää sille kehitettyä skriptikieltä, jossa kukin skriptitiedoston rivi suorittaa yhden komennon. Skriptin toiminnallisuus on laajennettavissa dll-lisäosilla, joko valmiilla sellaisilla tai itse koodatuilla. Valmis asennusskripti käännetään NSIS-kääntäjällä suoritettavaksi asennusohjelmaksi.

InstallAnywhere on Javaa käyttävä asennusohjelmisto, jolla voidaan toteuttaa ohjelmiston asennus useille eri käyttöjärjestelmille, kuten Linuxille, Mac OS X:lle ja Windowsille. Alkujaan Zero G Softwaren kehittämän ohjelmiston omistaa kirjoitushetkellä Acresso Software, jonka tuotteisiin kuuluu myös InstallShield-asennustyökalut. (InstallAnywhere Wikipedia 2008.)

InstallAnywherellä tehty asennusohjelma voidaan tehdä joko suoritettavaksi tiedostoksi tai puhtaaksi Java-sovellukseksi (.jar -tiedosto). Suoritustiedoston etuna on sen käynnistettävyys graafisessa ympäristössä hiirellä tai konsolissa tiedoston nimellä, Java-sovellus puolestaan on käynnistettävä kohdejärjestelmästä riippuvalla tavalla, useimmiten konsolista. (InstallAnywhere Evaluation Guide 2009)

Eri teknologioilla tehdyt asennukset tarjoavat käyttöliittymältään melko samantyyppisen asennuskokemuksen käyttäjälle, kaikkien sisältäessä tuen monikielisille asennuksille sekä ohjelmiston päivityksille. Eroavuudet löytyvät asennuksen suorittavista asennusmoottoreista, asennuksen tiedostokoosta ja tekoon käytettävistä kehitystyökaluista.

3.2 Windows Installer

3.2.1 Tausta

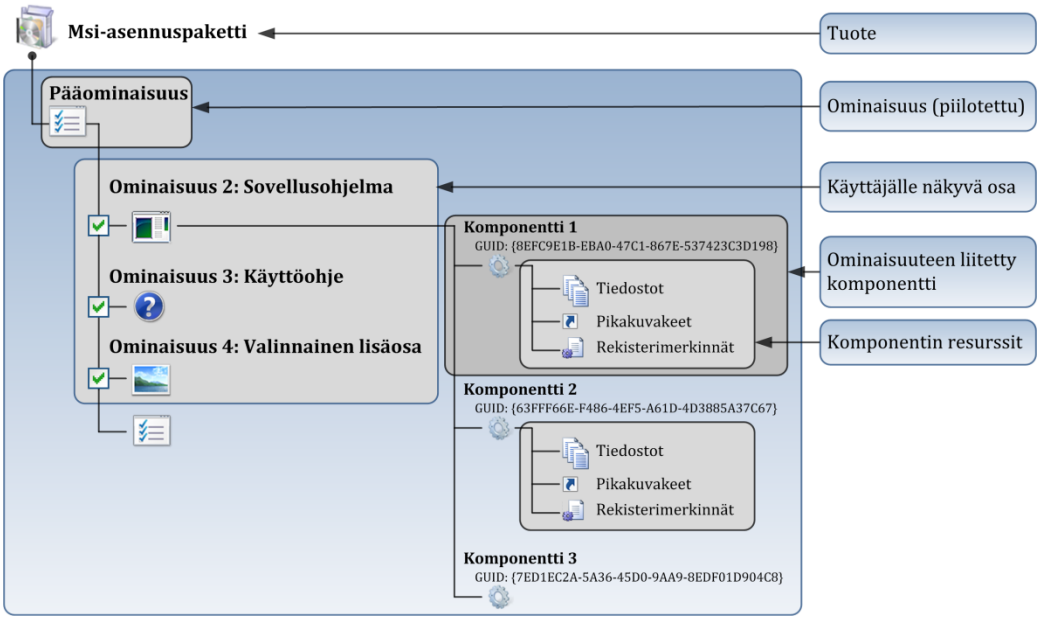
1990-luvulla Microsoft Officen tiimi halusi uudistaa Office-ohjelmiston asennuksen, sillä heidän tekniseen tukeensa tuli liikaa Officen asentamiseen tai poistamiseen liittyviä yhteydenottoja. Työnimellä *Darwin* perustetun projektin versiota 1.0 käytettiinkin *Microsoft Office 2000*:n asentamisessa. Uuden teknologian haluttiin olevan muidenkin ohjelmistovalmistajien käytettävissä ja sen kehittäminen siirtyi

NT5 tiimille, joka työsti Windows 2000 -käyttöjärjestelmää. Darwin nimettiin *Windows Installeriksi* ja siitä tuli keskeinen osa ohjelmistojen asennusmekanismia Windows-järjestelmissä osana Microsoftin *Zero Administration Windows* -aloitetta, jonka tarkoituksena oli vähentää työasemien kokonaiskustannuksia yrityksissä. Windows 2000 sisälsi kokonaan uuden lähestymistavan keskitetyille ohjelmistoasennuksille verkkoympäristössä. (Baker 2002, 83–84.)

Asennusta tehtäessä ohjelmiston järjestelmävaatimukset määrittävät Windows Installerin version, joka voidaan olettaa löytyvän käyttäjän koneelta ennen asennusta. Windows Installerin päivittäminen asennuksen aikana vaatii käytännössä ulkoisen käynnistäjän (*bootstrapper*) käyttöä, joka tarkistaa käytössä olevan Installerin version, päivittää sen tarvittaessa ja käynnistää sen jälkeen varsinaisen ohjelmiston asennuksen.

3.2.2 Asennuksen rakenne

Windows Installer -palvelu käsittelee ohjelmistoja tuotteina, asennusominaisuuksina (*feature)* ja komponentteina. Kuviossa 6 on esitetty yksinkertaistetusti tätä rakennetta.



KUVIO 6. Windows Installer -asennuksen yksinkertaistettu rakenne

Komponentti on pienin ja olennaisin edellä mainituista osatekijöistä. Se koostuu resursseista, kuten tiedostoista, rekisterimerkinnöistä, pikakuvakkeista tai muista asennettavista osista. Komponentin resursseja käsitellään aina yhtenä kokonaisuutena, joka joko asennetaan tai poistetaan. Resurssin on tarkoitus sisältyä vain yhteen komponenttiin, esimerkiksi samaa tiedostoa ei pitäisi olla kahden eri komponentin resurssina, vaikka komponentit kuuluisivatkin eri tuotteisiin. Eri tuotteiden asentaessa yhteisiä tiedostoja tuotteiden on sisällettävä sama yhteinen komponentti. (Windows Installer Service Overview 1999.)

Jokaiselle komponentille luodaan komponenttikoodi (*ComponentCode*), joka on komponentin globaalisti yksilöivä GUID-arvo. Yksi resursseista voidaan valita komponentin avaimeksi (*keypath*), jonka Windows Installer palauttaa kysyttäessä komponentin asennussijaintia. Avainta käytetään myös komponentin eheyden tarkistukseen. Mikäli avaimeksi määritettyä resurssia ei löydy, komponentti tulkitaan rikkinäiseksi. (Windows Installer Service Overview 1999.)

Ohjelmistoa asentava loppukäyttäjä ei näe komponentteja, niiden kanssa ovat tekemisissä pääasiassa vain asennuskehittäjät. Asennusominaisuudet sisältävät yhden tai useamman komponentin muodostaen ohjelmistosta toiminnallisia kokonaisuuksia. Ominaisuudet on usein järjestetty hierarkkiseksi rakenteeksi, jossa osa ominaisuuksista on piilotettu ja osa käyttäjän valittavissa. Ominaisuudet ovat tuotekohtaisia eikä niillä ole yksilöiviä tunnuksia. Komponentin ja ominaisuuden välillä ei ole omistussuhdetta, joten eri ominaisuudet voivat sisältää samoja komponentteja.

Tuotteen käsite Windows Installerin yhteydessä tarkoittaa yksittäistä msi-pakettia, joka sisältää yhden tai useamman ominaisuuden. Kuten komponenteilla, tuotteellakin on globaalisti yksilöivä GUID-tunniste, tuotekoodi (*ProductCode*), jonka avulla Windows Installer kykenee selvittämään, mitkä tuotteet ovat asentaneet tietyn komponentin. (Windows Installer Service Overview 1999.)

3.2.3 Tiedostotyypit

Windows Installeriin liittyy useita tiedostotyyppejä, joista tärkeimmät on esitelty taulukossa 1.

TAULUKKO 1. Windows Installerin tiedostotyyppejä (Windows Installer File Name Extensions 2008)

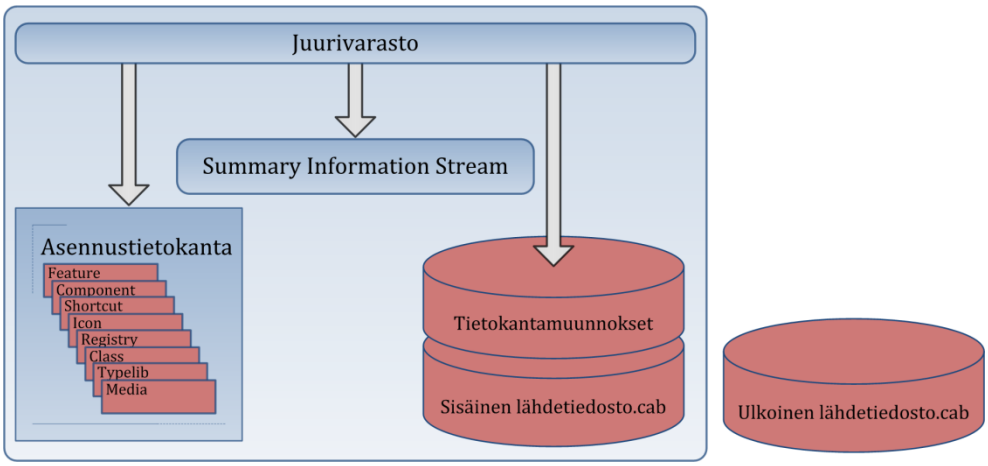
Tiedostyyppi	**Kuvaus**
.msi	Asennuspaketti. Tiedostotyyppi on assosioitu Windows Installerille, joten tiedosto avaus käynnistää asennuksen.
.msp	Windows installerin asennuksen päivityspaketti. Sisältää asennuksen päivitysten lisäksi tiedon, mihin asennuksen versioihin päivitys soveltuu.
.mst	Asennustietokannan muunnostiedosto. Asennustietokantaan tehdyt muutokset voidaan tallettaa mst-tiedostoon, jolloin itse asennustietokantaa ei muuteta. Muunnos otetaan käyttöön asennuksen käynnistysvaiheessa komentorivin kautta tai se voidaan upottaa msi-pakettiin sisäiseksi tietokantamuunnokseksi.
.msm	Merge-moduulitiedosto, joka on tarkoitettu asennustietokantojen yhdistämiseen. Moduuli sisältää yhden tai useamman asennuskomponentin sekä kaiken asennuslogiikan mitä komponenttien asentamiseen ja poistamiseen tarvitaan. Moduuli yhdistetään osaksi asennustietokantaa asennuksen tekovaiheessa ja siitä tulee kiinteä osa tietokantaa.

3.2.4 Asennuspaketti

Windows Installer -asennuspaketti on rakenteellisesti järjestetty tiedosto (COM Structured Storage file), tiedostojärjestelmä tiedoston sisällä. Tietokoneen hakemistoja vastaavat *varastot (storages)* ja tiedostoja *virrat (streams)*. Rakenteellisesti järjestetyllä tiedostolla on aina tiedoston nimen mukainen juurivarasto *(root strorage)*, johon alivarastot ja alivirrat liittyvät. Komponenttiobjektimalliin *(COM)* pohjautuvan rakenteellisesti järjestetyn tiedoston hierarkkinen rakenne on

standardoitu ja siten myös muiden COM-sovellusten luettavissa, mutta virtojen tulkitseminen on kuitenkin sovelluskohtaista. Poikkeuksena edelliseen on *Summary Information Stream*, jonka formaatti on tunnettu ja kaikkien sovellusten tulkittavissa. Summary Information sijaitsee aina välittömästi juurivaraston alla. (Baker 2001, 59–60.)

Msi-paketin rakenne on kuvion 7 mukainen, sen sisältäessä aina vähintään asennustietokannan sekä Summary Information -virran. Näiden lisäksi se voi sisältää upotettuja tietokantamuunnoksia sekä sisäisiä tai ulkoisia lähdetiedostoja. Sisäiset lähdetiedostot ovat msi-pakettiin upotettuja, CAB-pakattuja tiedostoja, ulkoiset puolestaan msi-paketin ulkopuolisia ja voivat CAB-pakattuja tai pakkaamattomia. (Baker 2002, 88.)



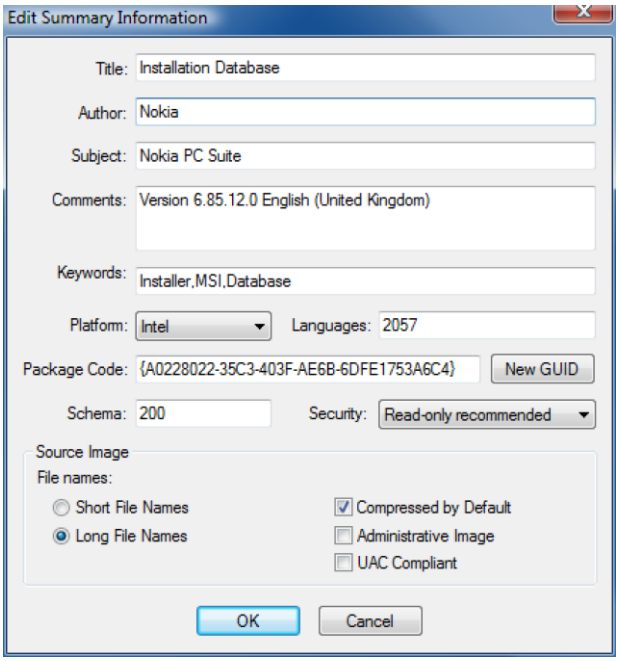
KUVIO 7. Msi-asennuspaketin rakenne

Summary Information sisältää 17 kenttää, joista neljällä on oltava määritettynä arvo msi-paketissa. Vaaditut kentät on selitetty taulukossa 2. Virran standardoinnin vuoksi sen kenttien nimillä ja tarkoituksilla msi-paketissa ei välttämättä ole yhteyttä. (Baker 2002, 94.)

TAULUKKO 2. Summary Informationissa vaaditut kentät (Summary Information Stream 2008)

<table>
<tr><th>Kenttä</th><th>Kuvaus</th></tr>
<tr><td>Revision Number</td><td>Asennuspaketin pakettikoodi, joka yksilöi paketin.</td></tr>
<tr><td>Page Count</td><td>Arvo identifioi pienimmän Windows Installer version, jolla asennuspaketti voidaan asentaa. Arvo 200 vastaa Installerin versiota 2.0.</td></tr>
<tr><td>Template</td><td>Järjestelmäalusta sekä asennuspaketin tukemat kielet muodossa: alusta;kieli;kieli… Esim. Intel;1033,2046</td></tr>
<tr><td>Word Count</td><td>Bittityyppinen kenttä, joka identifioi lähdetiedostojen ominaisuudet. Bitti 3 on käytössä Windows Vistassa Installer 4.0:sta alkaen.
<table>
<tr><th>Bitti</th><th>Arvo</th><th>Kuvaus</th></tr>
<tr><td>Bitti 0</td><td>0</td><td>Pitkät tiedostonimet</td></tr>
<tr><td></td><td>1</td><td>Lyhyet tiedostonimet (DOS 8.3 muoto)</td></tr>
<tr><td>Bitti 1</td><td>0</td><td>Lähdetiedostot pakkaamattomia</td></tr>
<tr><td></td><td>2</td><td>Lähdetiedostot pakattuja</td></tr>
<tr><td>Bitti 2</td><td>0</td><td>Lähdetiedostot ovat alkuperäisessä sijainnissaan</td></tr>
<tr><td></td><td>4</td><td>Lähdetiedostot kuuluvat hallinnollisella asennuksella tehtyyn levykuvaan.</td></tr>
<tr><td>Bitti 3</td><td>0</td><td>Paketin asentaminen voi edellyttää kohotettuja oikeuksia.</td></tr>
<tr><td></td><td>8</td><td>Paketin asentaminen ei edellytä kohotettuja oikeuksia.</td></tr>
</table></td></tr>
</table>

Kuviossa 8 on esitetty Msi-paketin Summary Information -ominaisuuskentät.



KUVIO 8. Summary Information

Windows Installerille Summary Informationilla on kaksi tarkoitusta. Ensimmäinen liittyy asennuspaketin ominaisuuskenttien tarkastelemiseen Windows Explorerin kautta ja toinen vaadittuihin ominaisuuksiin, joita tarvitaan paketin asentamisessa. (Summary Information Stream 2008.)

3.2.5 Asennustietokanta

Windows Installer sisältää yli 80 sisäänrakennettua tietokantataulua, jotka muodostavat asennuksen relaatiotietokannan. Yksittäisessä asennuksessa tarvittavien taulujen määrä vaihtelee ohjelmiston asentamiseen tarvittavista toimenpiteistä riippuen. Yleisessä tapauksessa tietokanta sisältää:

- ohjelmiston asennettavissa olevat ominaisuudet
- ohjelmiston asennuskomponentit
- ominaisuuksien ja komponenttien väliset relaatiot
- Windowsin järjestelmärekisteriin kirjoitettavat avaimet ja arvot

- pikakuvakkeen asennetun ohjelman käynnistämiseksi
- asennuksen käyttöliittymän
- ohjelmiston päivitykseen tarvittavat tiedot.

Tietokannan tärkeimmät taulut ryhmitellään seuraavasti:

- *Ydintaulut* (LIITE 1) sisältävät asennuksen perusominaisuudet ja komponentit. Ydintaulujen sisältö vaikuttaa muihin tauluihin, joten ne määritetään asennusta tehtäessä ensimmäisenä.
- *Tiedostotaulut* (LIITE 2) sisältävät asennukseen kuuluvat tiedostot ja niihin liittyvät toiminnot. Tiedostotaulujen sisältö määritetään ydintaulujen määrittelyn jälkeen.
- *Rekisteritaulut* (LIITE 3) sisältävät Windowsin järjestelmärekisteriin kirjoitettavan informaation. Erityyppisille rekisterimerkinnöille on varattu omat taulunsa, joita käyttämällä hyödynnetään Windows Installerin kehittyneempiä ominaisuuksia, kuten COM-palvelimien rekisteröintejä ja tiedostoassosiaatioiden luontia.
- *Etsintätauluja* (LIITE 4) käytetään tiedostojen ja sovellusten etsimiseen rekisteristä, Windows Installerin kirjanpidosta, kiintolevyn hakemistoista tai ini-tiedostoista.
- *Informaatiotaulut* (LIITE 5) sisältävät yleisiä asennuksen aikana tarvittavia tietoja.
- *Asennusproseduuritaulut* (LIITE 6) määrittävät asennuksen aikana suoritettavat standardit ja mukautetut toiminnot. Asennussekvenssitaulut liittyvät päätoimintoihin INSTALL, ADMIN tai ADVERTISE jonka alitoiminnot ne määrittävät. Päätoiminto asetetaan komentorivillä tai ohjelmallisesti käytettäessä Installerin API-funktioita.
- *Käyttöliittymätaulut* (LIITE 7) määrittävät Windows Installerin sisäänrakennetun graafisen käyttöliittymän. Käytettävissä on rajoitettu määrä standardeja Windows-käyttöliittymäkomponentteja, mm. tekstikenttä, vieritettävä tekstikenttä, paino- ja radionappi, lista- yhdistelmäruutu, valintaruutu, puu- ja listanäkymät, hakemistolista ja edistymispalkki. (Installer Database 2008.)

3.2.6 Standardit toiminnot (Standard Actions)

Windows Installer sisältää lukuisia sisäänrakennettuja, SDK:ssa dokumentoituja standardeja toimintoja. Useimmat ohjelmiston asennuksessa tarvittavat toimenpiteet on toteutettavissa standardeilla toiminnoilla ja niiden käyttäminen on yleensä suositeltavampaa kuin oman toteutuksen luominen standardin toiminnon korvaajaksi. Standardit toiminnot noudattavat Windows Installerin deklaratiivista toimintamallia ja toiminnon epäonnistuessa järjestelmä palautetaan toimintoa edeltävää tilaan. Standardeja toimintoja ovat esimerkiksi:

- *CreateShortcuts*, jolla luodaan pikakuvakkeita asennettuihin tiedostoihin
- *InstallFiles*, joka kopioi asennettavat tiedostot asennuspaketista asennushakemistoon
- *WriteRegistryValues*, joka kirjoittaa järjestelmärekisteriin rekisteritaulussa määritetyt arvot. (Standard Actions 2008.)

Standardit toiminnot lukevat tarvitsemansa tiedot asennustietokannan tauluista, joten niiden toimintaa ei tarvitse erikseen määritellä. Toimintoja käytetään joko sisällyttämällä ne asennussekvenssitauluihin tai käyttämällä Installerin API -funktiota *MsiDoAction*. Mikäli toimintoon liittyvissä tauluissa ei ole tietoja, sen suorittamisella ei ole vaikutusta. Asennustyökaluohjelmistot huolehtivat standardien toimintojen lisäämisestä sekvenssitauluihin ja huomioivat niiden mahdolliset suoritusjärjestysrajoitukset, joten asennuksen tekijän ei yleensä tarvitse huolehtia toimintojen mukanaolosta asennustietokannassa. SDK:n työkaluja suoraan käytettäessä asennuksen tekijän on huolehdittava myös näistä yksityiskohdista. Koska yksikertaisenkin ohjelmiston asennustietokannan muodostaminen ja ylläpitäminen käsityönä on varsin työläs ja virhealtis tehtävä, asennuksia ei käytännössä kannata tehdä pelkästään SDK:n tarjoamilla työkaluilla.

3.2.7 Mukautetut toiminnot (Custom Actions)

Tilanteissa, joissa standardi toiminto ei ole riittävä suorittamaan vaadittua toimenpidettä tai siihen ei ole olemassa standardia toimintoa, asennuksen tekijä voi määritellä mukautetun toiminnon. Mukautettuja toimintoja tarvitaan esimerkiksi:

- käynnistämään asennettu ohjelma asennuksen päätteeksi
- näyttämään ohjelmistoon liittyviä dokumentteja, kuten ohjelmistopäivityksen sisältämistä korjauksista kertova dokumentti tai ohjelmiston www-sivusto
- sulkemaan käynnissä olevia ohjelmia
- omien virheilmoitusten tai viestien näyttämiseen
- omien apuohjelmien tai dll-kirjastofunktioiden suorittamiseen. (Custom Actions 2008.)

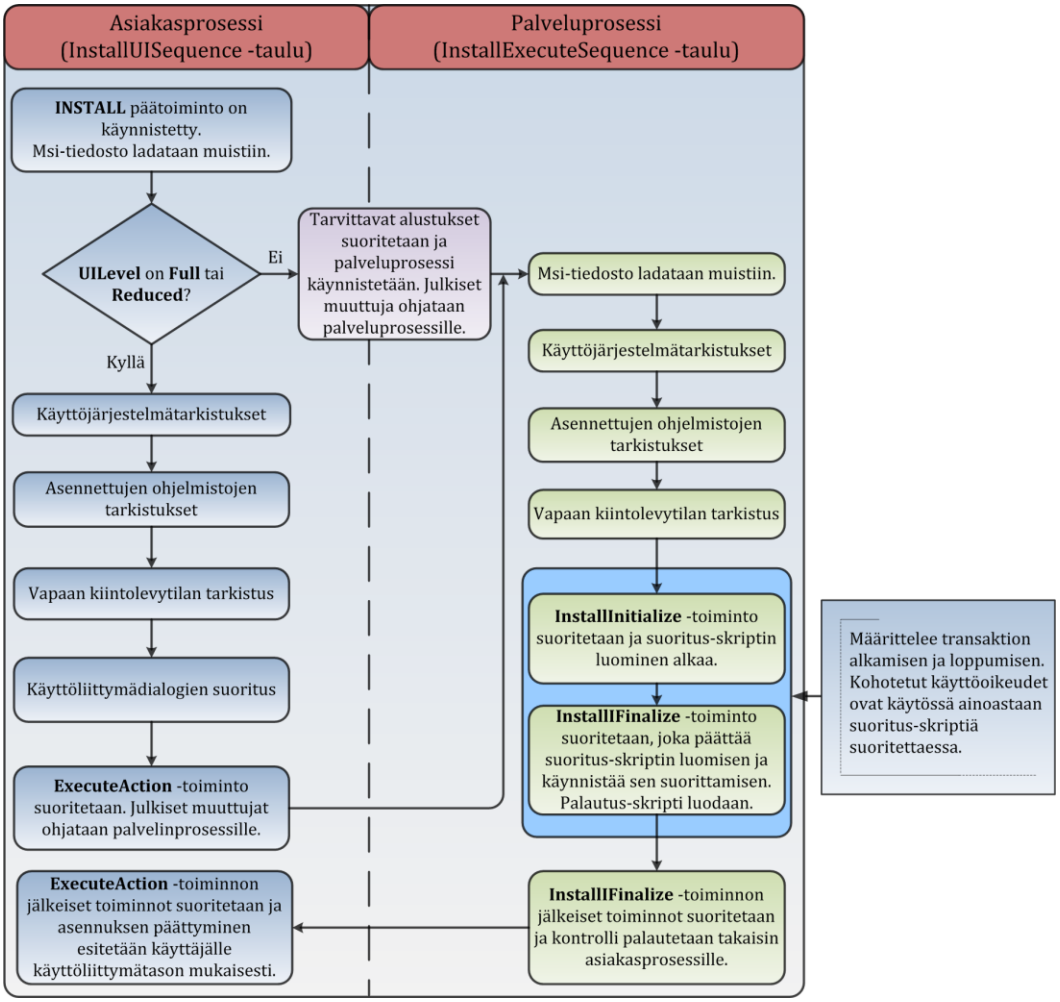
Mukautettu toiminto lisätään asennustietokantaan määrittämällä se CustomAction-tauluun ja lisäämällä luoto toiminto sekvenssitauluihin. Toiminnon tyypistä riippuen sen suorittava tiedosto:

- sijaitsee binääritaulussa, josta se kopioidaan ajon aikana kiintolevylle väliaikaiseen hakemistoon ja poistetaan suorituksen päätyttyä
- asennetaan ohjelmiston mukana
- on jo olemassa kohdejärjestelmässä.

Windows Installerin tukemat mukautetut toiminnot on listattu liitteessä 8. Mukautettu toiminto voi olla dll-kirjastofunktio, suoritettava exe-tiedosto, JScript- tai VBScript -tiedosto/komentosarja tai se voi asettaa ajonaikaisen tietokannan muuttujia. Oletuksena toiminto suoritetaan välittömästi sen sekvenssinumeron osoittamassa paikassa, mutta suoritus voidaan siirtää suoritus-skriptiin, jolloin toiminto voidaan tarvittaessa suorittaa kohotetuilla oikeuksilla järjestelmätilin alaisuudessa.

3.2.8 Asennusprosessi

Tyypillinen Windows Installer -asennusprosessi on kuvion 9 mukainen.



KUVIO 9. Windows Installer -asennusprosessi

Päätoiminto *INSTALL* asetetaan oletuksena, toiminnot *ADMIN* tai *ADVERTISE* pitää asettaa komentorivillä optioilla /a tai /j. Asennuksen käynnistyessä asiakasprosessi kopioi asennuspaketin yksilöllisellä nimellä käyttäjäkohtaiseen väliaikaishakemistoon ja asettaa julkisen muuttujan DATABASE osoittamaan tähän tiedostoon. Tämän jälkeen asiakasprosessi lataa asennustietokannan muistiin. (Baker 2004, 36–38.)

Käynnistyksen yhteydessä asetettava käyttöliittymän tason osoittava muuttuja *UILevel* tarkistetaan. Käyttöliittymätasot on kuvattu taulukossa 3.

TAULUKKO 3. Käyttöliittymätasot (User Interface Levels 2008)

Käyttoliittymätaso	**Kuvaus**
Täysi (Full UI) `UILevel = 5`	Asennustietokannan modaaliset ja ei-modaaliset dialogit sekä virheilmoitukset näytetään. Modaaliset dialogit vaativat käyttäjän toimia asennuksen jatkamiseksi, ei-modaaliset dialogit eivät. Täysi käyttöliittymä on yleensä toteutukseltaan velhotyylinen, sisältäen yhden tai useampia vaihteita.
Rajoitettu (Reduced UI) `UILevel = 4`	Asennustietokannan ei-modaaliset dialogit, virheilmoitukset sekä joitakin erityistilanteisiin liittyviä dialogeja näytetään.
Perus (Basic UI) `UILevel = 3`	Windows Installerin sisäänrakennetut, ei-modaaliset edistymisilmoitukset sekä virheilmoitukset näytetään. Asennustietokannan dialogeja ei näytetä.
Tyhjä (None) `UILevel = 2`	Täysin käyttöliittymätön asennus (silent).

Mikäli *UILevel* on perustasoa korkeampi, *InstallUISequence* -taulu käydään läpi ennen *InstallExecuteSequence* -taulua. Perustasolla tai alhaisemmalla *InstallUISequence* -taulu ohitetaan. Tällöin asiakasprosessi valmistelee palveluprosessin suorituksen ja ohjaa julkiset muuttujat sekä DATABASE-muuttujan arvon palveluprosessille. (Baker 2004, 38.)

Alustusten ja käyttöliittymätason tarkistuksen jälkeen asennuksen toimintojen suoritus alkaa. *LaunchConditions* -toiminto käy läpi *LaunchCondition* -taulun varmistaen asennuksen suorittamiseen liittyvien ehtojen täyttymisen. Mikäli jokin taulun ehdoista ei täyty, asennus keskeytetään ja käyttäjälle näytetään virheilmoitus. Suoritusehtojen tarkistuksen jälkeen järjestelmään jo asennettujen ohjelmistojen joukosta etsitään vaadittuja, ei-yhteensopivia, päivitettäviä tai muita ohjelmistoja etsintätauluja käyttävillä toiminnoilla *AppSearch, CCPSearch, RMCCPSearch*, sekä *Upgrade*-taulua käyttävällä *FindRelatedProducts* -toiminnolla. (Baker 2004, 38–39.)

Ohjelmistotarkistusten jälkeen, ennen käyttöliittymädialogien näyttämistä asennus tarkistaa oletuksena asennettavien komponenttien vaatiman kiintolevytilan. Laskennassa huomioidaan lisättävien ja poistettavien tiedostojen, rekisterimerkintöjen, ini-tiedostojen sekä pikakuvakkeiden vaatima kiintolevytila. Laskennassa ei huomioida asennuksen väliaikaista tilantarvetta, kuten asennuksen peruuttamiseen tarvittavien tiedostokopioiden käyttämää tilaa. (Baker 2004, 39–44.)

Käyttöliittymädialogien suorituksen aikana käyttäjän valinnat, kuten asennushakemisto ja asennettavaksi valitut ominaisuudet asetetaan. Käyttöliittymävaihe päättyy ei-modaalisen edistymisdialogin näyttämiseen, jolloin asiakasprosessi käynnistää *ExecuteAction* -toiminnon, joka puolestaan käynnistää palveluprosessin suorituksen. *ExecuteAction* -toiminto välittää muuttuneet julkiset muuttujat palveluprosessille. Asennustietokannassa määritettyjä julkisia muuttujia ei välitetä, jos niiden arvo ei muuttunut asiakasprosessissa, koska palveluprosessi avaa asennustietokannan omassa prosessissaan ja voi lukea muuttujat sieltä. (Baker 2004, 44–45.)

Kontrollin siirryttyä palveluprosessille se lataa asennustietokannan muistiin sekä kopioi sen ilman upotettuja lähdetiedostoja Installerin välimuistihakemistoon *%SystemRoot%\Installer*. Tämän jälkeen palvelu tekee kyselyn *InstallExecuteSequence* -tauluun suoritettavien toimintojen selvittämiseksi. (Baker 2004, 45.)

Käyttöjärjestelmätarkistukset sekä asennettujen ohjelmistojen tarkistukset, jotka on kuviossa 9 mainittu molemmissa prosesseissa, suoritetaan palveluprosessissa vain, jos *InstallUISequence* -taulua ei käsitelty asiakasprosessissa (UILevel $\leq$ 3). Vapaan kiintolevytilan tarkistus ja muut laskennalliset toiminnot suoritetaan uudestaan palveluprosessissa, koska käyttäjän valinnat ovat saattaneet muuttaa aiempien laskelmien tuloksia. (Baker 2004, 46.)

InstallInitialize -toiminto määrittää järjestelmää muokkaavien toimintojen aloituskohdan. Siitä eteenpäin suoritettavat toiminnot aina *InstallFinalize* -toimintoon asti kuuluvat asennustransaktioon, joka voi tehdä muutoksia järjestelmään ja joka on peruutettavissa virhetilanteessa tai käyttäjän toimesta. (Baker 2004, 46.)

Windows Installer käy transaktioon kuuluvat toiminnot läpi ja lisää järjestelmää muokkaavat toiminnot suoritus-skriptiin, joka tallennetaan piilotettuun hakemistoon. Välittömästi suoritettaviksi merkityt toiminnot suoritetaan niiden osuessa kohdalle, eikä niitä lisätä skriptiin. Vain suoritus-skriptiin lisättyjä toimintoja voidaan suorittaa kohotetuin käyttöoikeuksin. (Baker 2004, 46–47.)

InstallInitialize -toiminto sijoittuu *InstallExecuteSequence* -taulun alkupuolelle ja *InstallFinalize* -toiminto sen loppuun, kuten kuviosta 10 ilmenee.

```
LaunchConditions  100
AppSearch  400
CCPSearch NOT Installed 500
RMCCPSearch NOT Installed 600
...
InstallInitialize  1500
...
RemoveRegistryValues  2600
RemoveShortcuts  3200
RemoveFiles  3500
...
InstallFiles  4000
CreateShortcuts  4500
WriteRegistryValues  5000
...
InstallFinalize  6600
...
```

KUVIO 10. Toimintojen sijoittuminen sekvenssiin

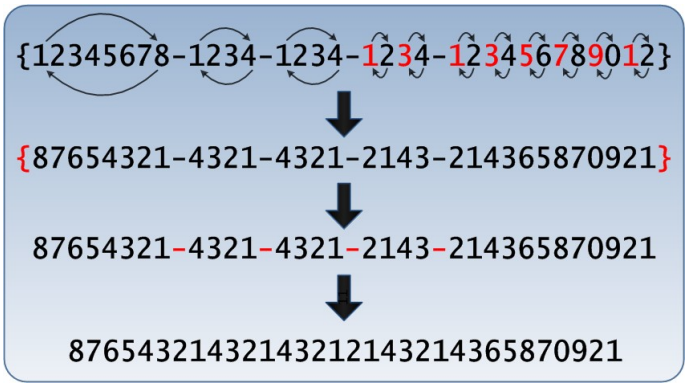
InstallFinalize -toiminto päättää suoritus-skriptiin luontiproseduurin ja käynnistää skriptin suorituksen. Jokainen suoritettu rivi lisätään palautus-skriptiin, joka ajetaan, mikäli asennus peruutetaan käyttäjän toimesta tai virhetilanteessa. Palautus peruuttaa muutokset palauttaen järjestelmän takaisin transaktiota edeltävään tilaan. *InstallFinalize* -toiminto päättyy, kun suoritus-skripti on onnistuneesti suoritettu. Palautus ei tämän jälkeen enää ole mahdollista. (Baker 2004, 47.)

InstallExecuteSequence -taulun toimintojen suorituksen jälkeen palvelu palauttaa kontrollin takaisin asiakasprosessille ja loput *InstallUISequence* -taulun toiminnot suoritetaan. (Baker 2004, 47.)

3.2.9 Asennettujen ohjelmistojen hallinta

Windows Installer käyttää globaalisti yksilöiviä tunnuksia (GUID) asennuspaketin, tuotteen, tuoteperheen ja komponenttien tunnuksina. Vastaavat muuttujat asennuspaketissa ovat pakettikoodi (*PackageCode*), tuotekoodi (*ProductCode*), päivityskoodi (*UpgradeCode*) ja komponenttikoodi (*ComponentCode*). Asennettaessa tuotetta edellä mainitut muuttujat talletetaan rekisteriin paketoidussa muodossa rekisterihakujen tehostamiseksi. Standardi GUID on 38 merkkiä pitkä, esimerkiksi: *{F5F6AA07-4D07-439A-BA69-95CA7C4A692D}*.

Paketoidun GUIDin pituus on 32 merkkiä. Paketointi tehdään kuvion 11 osoittamalla tavalla vaihtamalla merkkien järjestystä ja poistamalla aaltosulkeet sekä väliviivat (Baker 2002, 798). Edellä esitetty GUID paketoituna on siten: *70AA6F5F70D4A934AB9659ACC7A496D2*.



KUVIO 11. Standardin GUIDin muuntaminen paketoituun muotoon

Paketoidusta GUIDista edelleen tiivistetty muoto on *pakattu GUID*, jota Windows Installer käyttää vähentääkseen rekisteriin kirjoitettavan datan määrää. Pakattu GUID on pituudeltaan 20 merkkiä: *vUpAVO(8A$!!!!!MKKSk*. (Baker 2002, 798.)

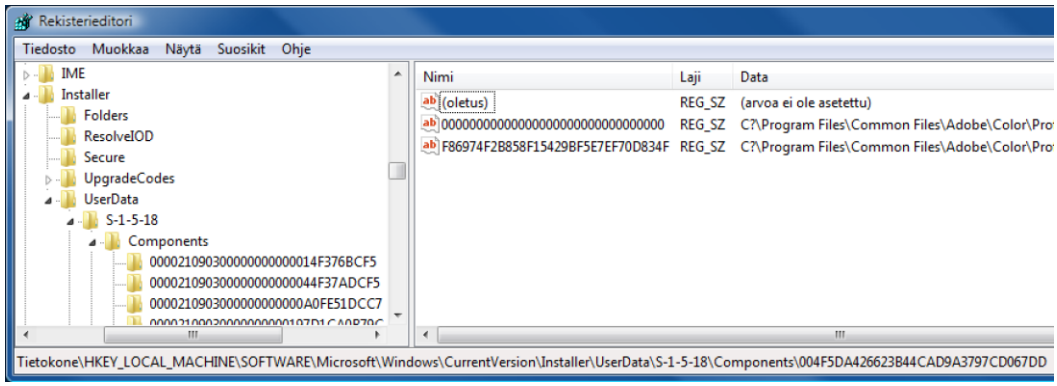
Pakatun GUIDin muodostamiseen käytettävää algoritmia ei ole kerrottu Windows Installerin dokumentaatiossa, mutta sen pääasiallinen käyttötarkoitus on *Darwin descriptorin* muodostaminen. Darwin descriptor on tuotekoodin, asennusominaisuuden ja komponenttikoodin yhdistelmä. Sillä on 4 erilaista esitystapaa asen-

nukseen sisältyvien asennusominaisuuksien ja komponenttien määrästä riippuen. Yleisin esitystapa on: *{Pakattu tuotekoodi}{Ominaisuus}>{Pakattu komponenttikoodi}*. Aaltosulkeet on merkitty esimerkkiin selkeyttämään eri osia, eivätkä kuulu varsinaiseen Darwin descriptoriin, kuten:
vUpAVO(8A$!!!!!MKKSkOUTLOOKFilesIntl_1033>FGJD,wN@VAR~9A'!lj$a
(Baker 2002, 799.)

Windows Installer pitää kirjaa asennetuista ohjelmistotuotteista, niiden ominaisuuksista, komponenteista ja päivityksistä järjestelmärekisterin avulla. Rekisteriä muokkaavat standardit toiminnot voidaan jakaa kolmeen ryhmään: yleisiä tuotetietoja muokkaavat toiminnot, komponenttitietoja muokkaavat toiminnot sekä tuotteiden ja asennusominaisuuksien tilatietoja muokkaavat toiminnot (Baker 2001, 303–306.).

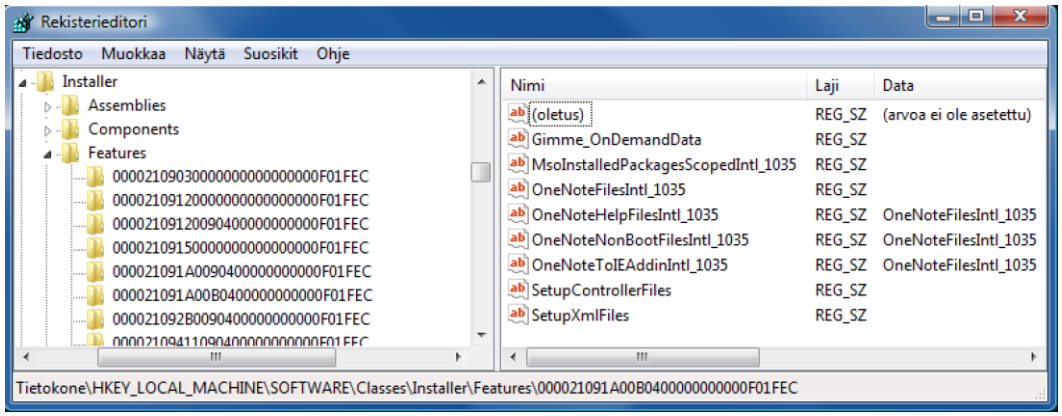
RegisterProduct -toiminto kirjoittaa tuotetietoja kuten julkaisijan, version, arvioidun tilankäytön ja ohjelmiston www-linkin rekisteriavaimeen: *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\tuotekoodi*, jossa *tuotekoodi* on msi-paketin tuotekoodi standardissa GUID-muodossa. Windowsin ohjauspaneelista löytyvä ohjelmien lisäys/poisto-ohjelma näyttää edellä mainitun *Uninstall*-avaimen alaisuuteen listatut tuotteet ja niiden ominaisuudet.

ProcessComponents -toiminto kirjaa konekohtaisesti asennetut komponentit rekisteriavaimeen: H*KEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Installer\UserData\S-1-5-18\Components*, jossa avain *S-1-5-18* tarkoittaa paikallisen järjestelmän tiliä, käyttäjäkohtaisessa asennuksessa komponentit kirjattaisiin käyttäjän tiliä vastaavan avaimen alaisuuteen. Kuvion 12 mukaisesti Components-avaimen aliavaimet ovat paketoituja komponenttikoodeja sisältäen arvoinaan tuotteet, jotka ovat asentaneet kyseisen komponentin sekä polun, johon komponentin pääavain osoittaa. Kuviossa 12 näkyvä, ainoastaan nollia sisältävä tuotekoodi tarkoittaa, että komponentti on asennettu pysyvästi järjestelmään, eikä Windows Installer poista sitä, vaikka sen asentanut tuote poistettaisiin.



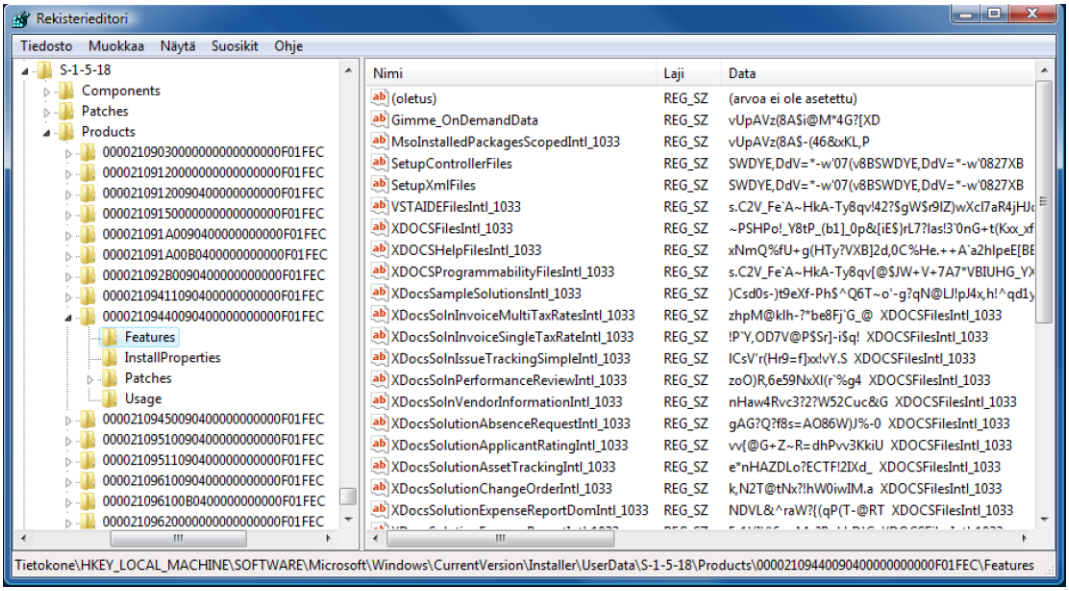
KUVIO 12. Windows Installerin asentamat komponentit rekisterissä

Tuotteen ja asennusominaisuuksien tilatietoja käsittelevät toiminnot *PublishProduct*, *PublishFeatures* ja *UnpublishFeatures* määrittävät tuotteen tai asennusominaisuuden tilan sekä ominaisuuksien isä-lapsi suhteet. *PublishFeatures* kirjaa asennusominaisuudet konekohtaisessa asennuksessa rekisteriavaimeen: *HKEY_LOCAL_MACHINE\SOFTWARE\Classes\Installer\Features* ja käyttäjäkohtaisessa asennuksessa avaimeen: *HKEY_CURRENT_USER\Software\Microsoft\Installer\Features*. Molemissa tapauksissa *Features*-avaimen aliavaimina on paketoituja tuotekoodeja sen mukaisesti mitä ohjelmistoja on asennettu. Kunkin asennetun tuotteen asennusominaisuudet ja niiden isä-ominaisuudet listataan arvoina tuoteavaimen alaisuuteen. Kuviossa 13 on esitetty osa konekohtaisesti asennettujen tuotteiden ominaisuuksista.



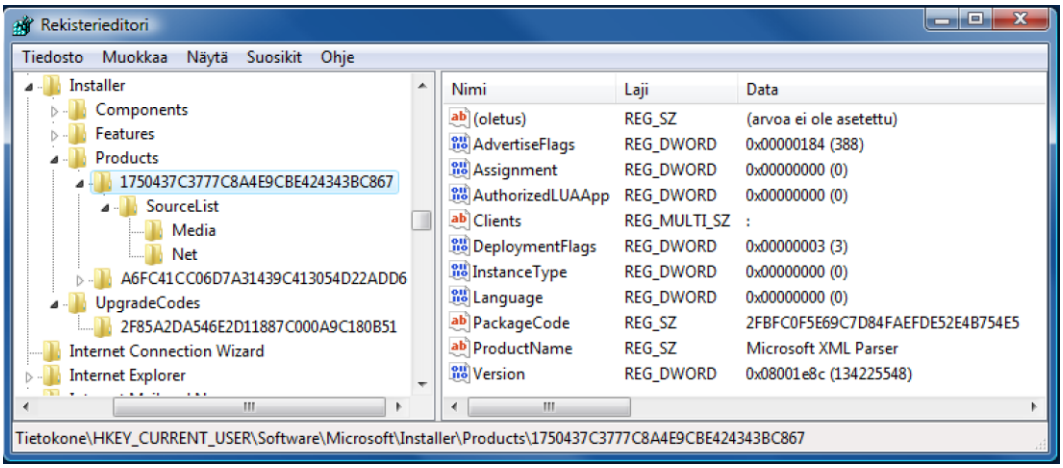
KUVIO 13. Asennettujen tuotteiden ominaisuuksia

Ominaisuuksiin kuuluvat komponentit kirjataan rekisteriavaimeen: *HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Installer\UserData\[SID]\Products\[paketoitu_tuotekoodi]\Features. Features* -avaimen arvojen nimet ovat ominaisuuksien nimiä, joiden tietosisältönä on lista asennusominaisuuteen liittyvistä komponenteista pakattuina komponenttikoodeina. Lapsiominaisuuden ollessa kyseessä listan päättää ominaisuuden isäominaisuuden nimi. Kuvio 14 havainnollistaa tätä rakennetta.



KUVIO 14. Asennusominaisuudet rekisterissä

PublishProduct -toiminto kirjoittaa käyttäjäkohtaisesti asennetun tuotteen asennustilan rekisteriavaimeen: *HKEY_CURRENT_USER\Software\Microsoft\Installer\Products\[paketoitu_tuotekoodi]* ja konekohtaisesti asennetun avaimeen: *HKEY_LOCAL_MACHINE\SOFTWARE\Classes\Installer\Products\[paketoitu_ tuotekoodi]*. Kuvio 15 havainnollistaa tuotteen tilaa kuvaavia arvoja. Kullekin tuotteelle luodaan myös *SourceList* -aliavain, joka sisältää alkuperäisen asennuspaketin sijainnin.



KUVIO 15. Asennetun tuotteen tila-arvoja

3.2.10 Ohjelmiston päivittäminen

Windows Installerilla asennettu ohjelmisto päivitetään joko uudella asennuspaketilla tai päivityspaketilla. Päivityspaketti (*patch*) sisältää ainoastaan vanhan ja uuden asennuspaketin väliset muutokset ja on kooltaan asennuspakettia huomattavasti pienempi, joten sen lataus verkosta ja asentaminen on täyttä asennusta nopeampaa. Riippumatta siitä kummalla tavalla päivitys jaetaan loppukäyttäjille, uusi asennuspaketti on aina luotava ja käytettävä päivitystyyppi asettaa sille tiettyjä ehtoja. Päivitystyyppejä on kolme:

- *Korjauspäivitys (Small Update).* Korjauspäivitystä voidaan käyttää tilanteissa, joissa uusi asennuspaketti sisältää niin pieniä muutoksia (esimerkiksi vaihdetaan yksi tiedosto), ettei asennuspaketin versiota tarvitse kasvattaa. Vain pakettikoodi vaihdetaan. Koska versiota ei kasvateta, asennettua päivitystä ei voi havaita muuten kuin vaihtuneista tiedostoista. Mikäli ohjelmiston halutaan olevan edelleen päivitettävissä tekemättä vielä tuotepäivitystä, korjauspäivityksen sijasta tulee käyttää versiopäivitystä, jolloin päivitysten järjestys tiedetään.
- *Versiopäivitys (Minor Upgrade).* Versiopäivityksessä uuden asennuspaketin versiota kasvatetaan ja pakettikoodi vaihdetaan. Versiopäivitys voi lisätä asennukseen uusia ominaisuuksia, mutta ei muokata tai poistaa olemassa olevia ominaisuuksia.

- *Tuotepäivitys (Major Upgrade).* Tuotepäivitys voi poistaa aiemman version ja kaikki siihen asennetut päivitykset. Asennuspaketin tuotekoodi, versio sekä pakettikoodi vaihdetaan. Tuotepäivitys vastaa periaatteessa tilannetta, jossa käyttäjä poistaa ohjelmiston itse ja asentaa uudemman version sen jälkeen. Päivitystapauksessa Windows Installer kuitenkin välittää poiston suorittavalle instanssille tiedon, että kyseessä on päivityksen yhteydessä tapahtuva ohjelmiston poistaminen. Asennuksen tekijä voi käyttää tätä tietoa hyväksi, joten poistotoiminnallisuudessa voi olla eroavuuksia riippuen siitä, poistetaanko ohjelmisto tuotepäivityksen yhteydessä vai itsenäisesti.

Korjaus- ja versiopäivityksissä päivitys perustuu ohjelmiston uudelleen asentamiseen. Asennuksen tuotekoodia ei vaihdeta ja päivitys vaatii asennuskomponentteihin liittyvien sääntöjen huolellista noudattamista, jotta asennustietokanta säilyttää eheytensä päivitysten välillä. Seuraukset eheyden rikkoutumisesta näkyvät useimmiten orvoiksi jääneinä resursseina, joita ei poisteta ohjelmiston poistossa.

Asennuspakettia käytettäessä päivityksen käynnistämiseksi Windows Installerille on välitettävä, joko komentorivillä tai ohjelmallisesti, tieto uudelleen asennettavista ominaisuuksista sekä tieto siitä, että Installerin välimuistiin talletettu asennustietokanta tulee korvata päivitysasennuksen tietokannalla. Komentorivi päivitykselle on: *msiexec /I [polku päivitettyyn asennuspakettiin] REINSTALL=ALL REINSTALLMODE=vomus*. Muuttujalla *REINSTALL* asetetaan uudelleen asennettavat ominaisuudet ja muuttujalla *REINSTALLMODE* uudelleenasennuksen toimenpiteet, jotka on selitetty taulukossa 4.

TAULUKKO 4. Uudelleenasennuksen optiot

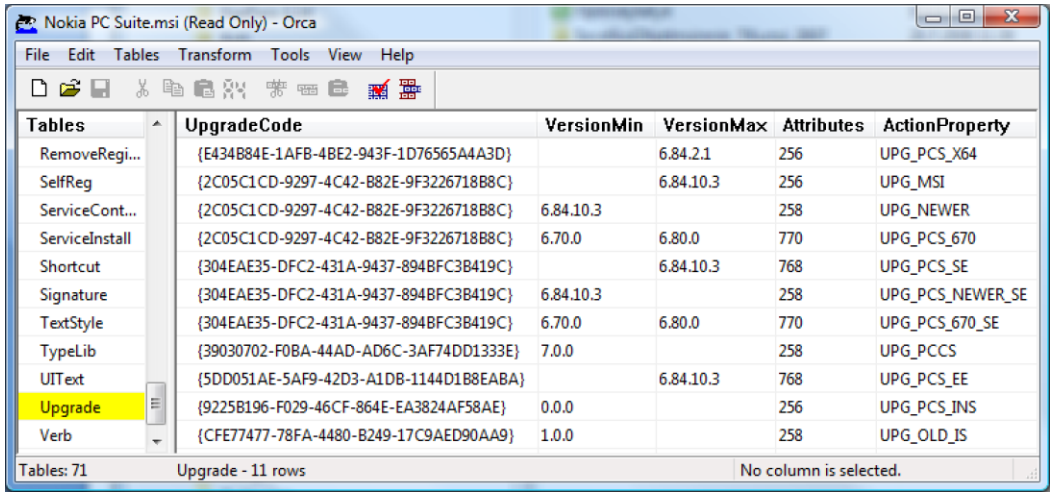
Optio	Kuvaus
p	Asennetaan uudelleen vain tiedoston puuttuessa.
o	Asennetaan uudelleen tiedoston puuttuessa tai jos asennettu versio on pienempi.
e	Asennetaan uudelleen tiedoston puuttuessa tai jos asennettu versio on yhtä suuri tai pienempi.
d	Asennetaan uudelleen tiedoston puuttuessa tai jos asennettu versio on eri suuri.
c	Tarkistussumma varmistetaan ja tiedosto asennetaan uudelleen sen puuttuessa tai jos tarkistussumma ei täsmää. Toimiakseen optio edellyttää, että asennuspakettia tehtäessä tiedostolle on luotu tarkistussumma tiedostotauluun.
a	Pakottaa kaikkien tiedostojen uudelleen asentamisen versiosta tai tarkistussummasta riippumatta.
u	Rekisteripolkuihin `HKEY_CURRENT_USER` ja `HKEY_USERS` asennettavat rekisteritaulun arvot asennetaan uudelleen.
m	Rekisteripolkuihin `HKEY_LOCAL_MACHINE` ja `HKEY_CLASSES_ROOT` asennettavat rekisteritaulun arvot asennetaan uudelleen.
s	Pikakuvakkeet ja välimuistiin talletetut kuvakkeet asennetaan uudelleen
v	Pakottaa Windows Installerin käyttämään uutta asennuspakettia välimuistiin talletetun paketin sijaan ja korvaamaan välimuistiin talletetun asennuspaketin uudella paketilla.

Ohjelmallisesti päivitettäessä sovelluksesta kutsutaan Windows Installerin API-funktiota *MsiReinstallProduct* ja välitetään funktioparametreilla päivitykseen tarvittavat tiedot.

Koska päivitysasennuksen käynnistäminen oikeaan tilaan vaatii komentoriviparametrien käyttöä, asennuspakettien jakaminen sellaisenaan ei sovellu jaettavaksi loppukäyttäjille. Tästä syystä msi-asennuspakettien kanssa käytetään monesti asennuksen käynnistämiseen tarkoitettua sovellusta (*bootstrapper*), jonka tiedostonimi on usein *setup.exe*. Se käynnistää asennuksen oikeaan tilaan ja voi sisältää myös muita asennusta edeltäviä toimintoja. Msp-päivityspaketteja käytettäessä

ongelmaa ei ole, koska päivityspakettia ei voi asentaa ellei päivityksen kohteena olevaa tuotetta ole asennettu. Msp-tiedostotyyppi on myös assosioitu Windows Installerille komennolla *"%SystemRoot%\System32\msiexec.exe" /p "%1" %**, joten msp-tiedoston avaaminen käynnistää päivityksen asentamisen.

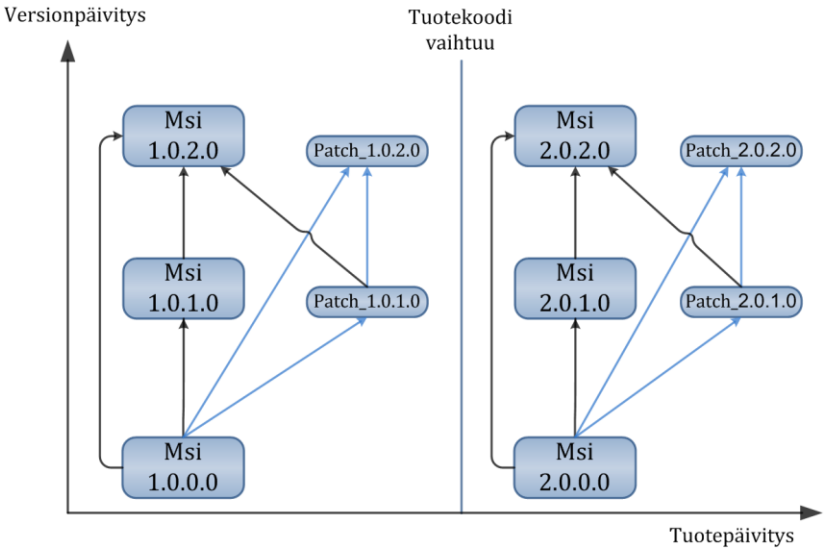
Tuotepäivityksessä asennuksen tuotekoodi vaihtuu, joten Windows Installer tulkitsee asennuspaketin uudeksi tuotteeksi, jota ei ole asennettu järjestelmään. Aiempia versioita ei automaattisesti poisteta, vaan poistettavat tuotteet on määriteltävä asennustietokannan *Upgrade*-taulussa (kuvio 16).



Tables	UpgradeCode	VersionMin	VersionMax	Attributes	ActionProperty
RemoveRegi...	{E434B84E-1AFB-4BE2-943F-1D76565A4A3D}		6.84.2.1	256	UPG_PCS_X64
SelfReg	{2C05C1CD-9297-4C42-B82E-9F3226718B8C}		6.84.10.3	256	UPG_MSI
ServiceCont...	{2C05C1CD-9297-4C42-B82E-9F3226718B8C}	6.84.10.3		258	UPG_NEWER
ServiceInstall	{2C05C1CD-9297-4C42-B82E-9F3226718B8C}	6.70.0	6.80.0	770	UPG_PCS_670
Shortcut	{304EAE35-DFC2-431A-9437-894BFC3B419C}		6.84.10.3	768	UPG_PCS_SE
Signature	{304EAE35-DFC2-431A-9437-894BFC3B419C}	6.84.10.3		258	UPG_PCS_NEWER_SE
TextStyle	{304EAE35-DFC2-431A-9437-894BFC3B419C}	6.70.0	6.80.0	770	UPG_PCS_670_SE
TypeLib	{39030702-F0BA-44AD-AD6C-3AF74DD1333E}	7.0.0		258	UPG_PCCS
UIText	{5DD051AE-5AF9-42D3-A1DB-1144D1B8EABA}		6.84.10.3	768	UPG_PCS_EE
Upgrade	{9225B196-F029-46CF-864E-EA3824AF58AE}	0.0.0		256	UPG_PCS_INS
Verb	{CFE77477-78FA-4480-B249-17C9AED90AA9}	1.0.0		258	UPG_OLD_IS

KUVIO 16. Upgrade-taulu

Tauluun listataan niiden tuotteiden päivityskoodit (*UpgradeCode*) ja versiot, joita Windows Installerin halutaan etsivän *FindRelatedProducts* -toiminnon aikana. Hakuehdot täyttävän tuotteen löytyessä sen tuotekoodi talletetaan riville määriteltyyn *ActionProperty* -muuttujaan. Mikäli rivin *Attibutes*-sarakkeen arvossa on mukana löydettyjen tuotteiden poistaminen, ne poistetaan *RemoveExistingProducts* -toimintoa suoritettaessa.



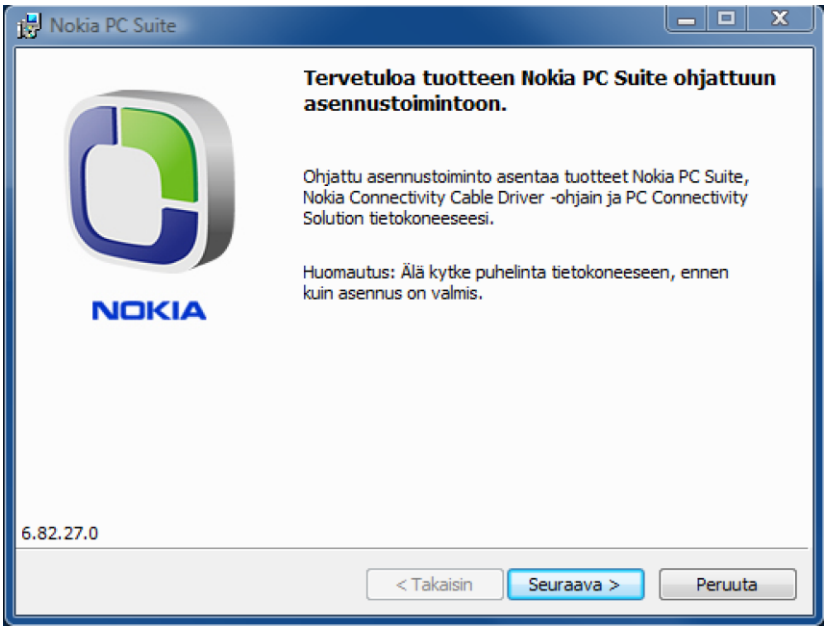
KUVIO 17. Esimerkki ohjelmistopäivityksestä

Kuviossa 17 on esitetty esimerkki ohjelmiston päivityksestä. Siinä ohjelmistosta on tehty kaksi tuoteversiota 1.0, ja 2.0. Kullekin tuoteversiolle on tehty kaksi versiopäivitystä. Esimerkin tilanne on ideaalinen tuotekoodin ja pääversionumeron (versionumeron ensimmäinen kenttä) vaihtuessa samaan tahtiin. Käytännössä tuotekoodin vaihtaminen on sidottu asennustietokannan yhteensopivuuteen asennuspakettien välillä ja tuotekoodi on vaihdettava, mikäli uusi tietokanta ei ole yhteensopiva aiempien versioiden kanssa. Tämä voisi tarkoittaa tuotekoodin vaihtoa jo aiemmin kuin versiossa 2.0. Versioinnissa huomioitavaa on se, että *FindRelatedProducts* tarkistaa versionumerosta vain kolme ensimmäistä kenttää, standardin Windows versionumeroinnin sisältäessä neljä kenttää.

4 NOKIA INSTALLER

4.1 Tausta

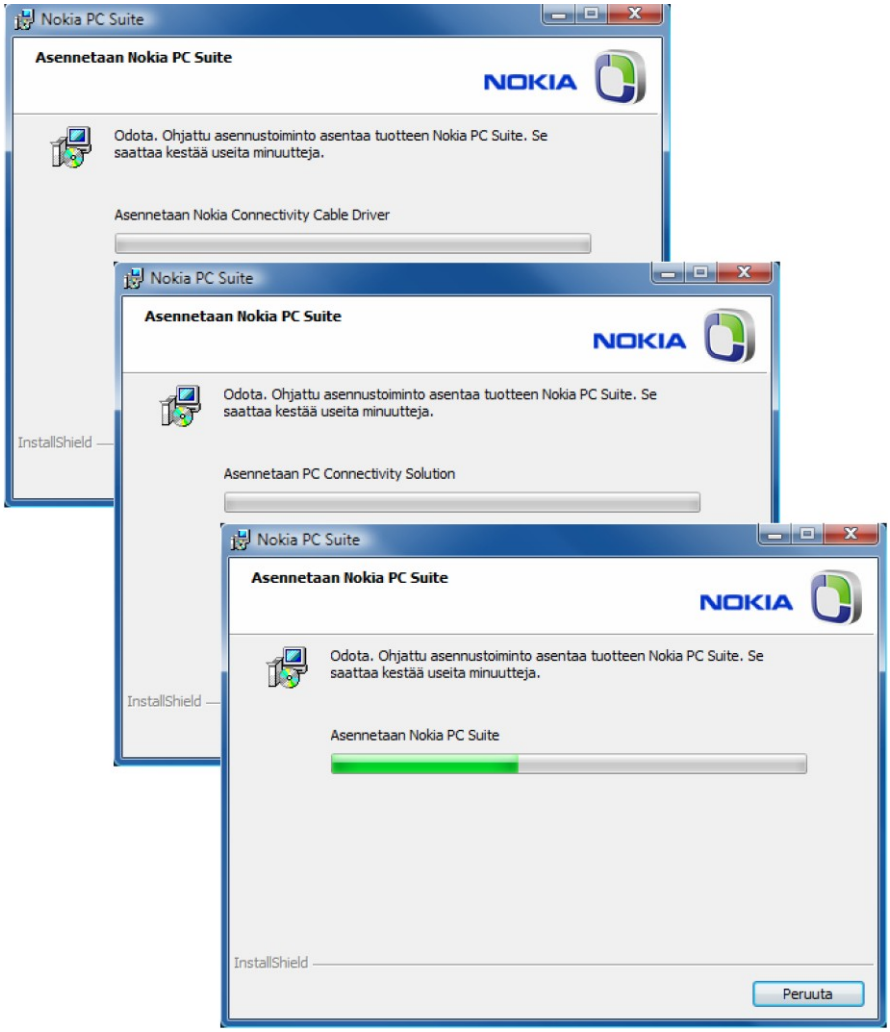
Nokia PC Suiten asennus sisältää PC Suite -sovellusten lisäksi PC:n ja matkapuhelimen kommunikoinnista vastaavan osan (PCCS) sekä laiteajurit puhelimen yhdistämiseksi tietokoneeseen USB-kaapelilla (CCD). Osat ovat erillisiä, itsenäisiä kokonaisuuksia ja ne asennetaan omista asennuspaketeistaan. PC Suiten käyttöönotto on kuitenkin haluttu pitää käyttäjille yksinkertaisena, joten asennettavaksi tarjottu asennuspaketti sisältää kaikki osat, kuten kuvion 18 esittämässä PC Suiten asennuksen aloitusdialogissa on esitetty.



KUVIO 18. Nokia PC Suiten msi-asennus

Ennen Nokia Installeria PCCS ja CCD upotettiin PC Suiten msi-pakettiin, josta ne purettiin ja asennettiin PC Suiten asennuksen toimesta UI-jaksossa ennen suoritusjakson alkua. Ratkaisu oli periaatteessa toimiva, mutta kuvion 19 mukaisesti asennuksen edistymisilmaisinta ei voitu kasvattaa PCCS:n ja CCD:n asennuksen

aikana, joten käyttäjälle välittyvä informaatio asennuksen edistymisestä oli puutteellinen. Lisäksi ratkaisu asetti rajoituksia asennusvaihtoehdoille, joista hiljaista asennusta (Windows Installerin komentorivin valitsin /q) ei voitu käyttää, koska silloin UI-jaksoa ei olisi suoritettu lainkaan ja PCCS sekä CCD olisivat jääneet asentamatta.



KUVIO 19. PC Suiten msi-asennuksen edistymisdialogi

Uusilla kehitystyökaluilla, kuten Microsoftin Visual Studio 2005:llä tai 2008:lla kehitetyt C++ -sovellukset ovat usein riippuvaisia Visual C++ -ajonaikaisista kirjastoista ja riippuvuudet on asennettava ennen ohjelmiston käyttöä. Visual C++ -kirjastoista on saatavana erilliset asennuspaketit sekä Windows Installer merge -moduulit, joita käytettäessä ajonaikaiset kirjastot tulevat kiinteäksi osaksi asennettavaa msi-asennuspakettia. Koska kirjastojen komponentit on toteutettu globaa-

leina *assembly-komponentteina*, Windows Installer paljastaa ne järjestelmälle vasta, kun asennuksen transaktio on suoritettu onnistuneesti. Tästä syystä kirjastojen käyttäminen asennuksen aikana, esimerkiksi kirjastoja käyttävän COM -palvelimen rekisteröinti voi johtaa virhetilanteeseen asennuksen aikana tai myöhemmin asennettua sovellusta käytettäessä. Ratkaisuna on asentaa riippuvuudet erillisistä asennuksista ennen niitä käyttävien sovellusten asentamista, joko käyttäen valmiita levitykseen tarkoitettuja asennuspaketteja (*vcredistx86.exe* ja *vcredistx64.exe*) tai rakentamalla merge-moduulien avulla kirjastoille omat msi-asennukset. Koska PC Suiten asennus sisälsi jo kaksi muuta upotettua msi-asennusta ja upotettujen asennusten aiheuttamat ongelmat tiedettiin, ulkoinen ratkaisu oli välttämätön.

Ohjelmistojen pilkkominen pienempiin osakokonaisuuksiin, joilla kullakin on oma asennuksensa tai muuten erillisten asennusten yhdistäminen yhdeksi kokonaisuudeksi vaatii Windows Installer 4.5:tä edeltävien versioiden kanssa ulkoisen käsittelijän, joka vastaa asennuspakettien välittämisestä Windows Installerille asennettavaksi. Tähän tarkoitukseen suunniteltiin ja toteutettiin Nokia Installer.

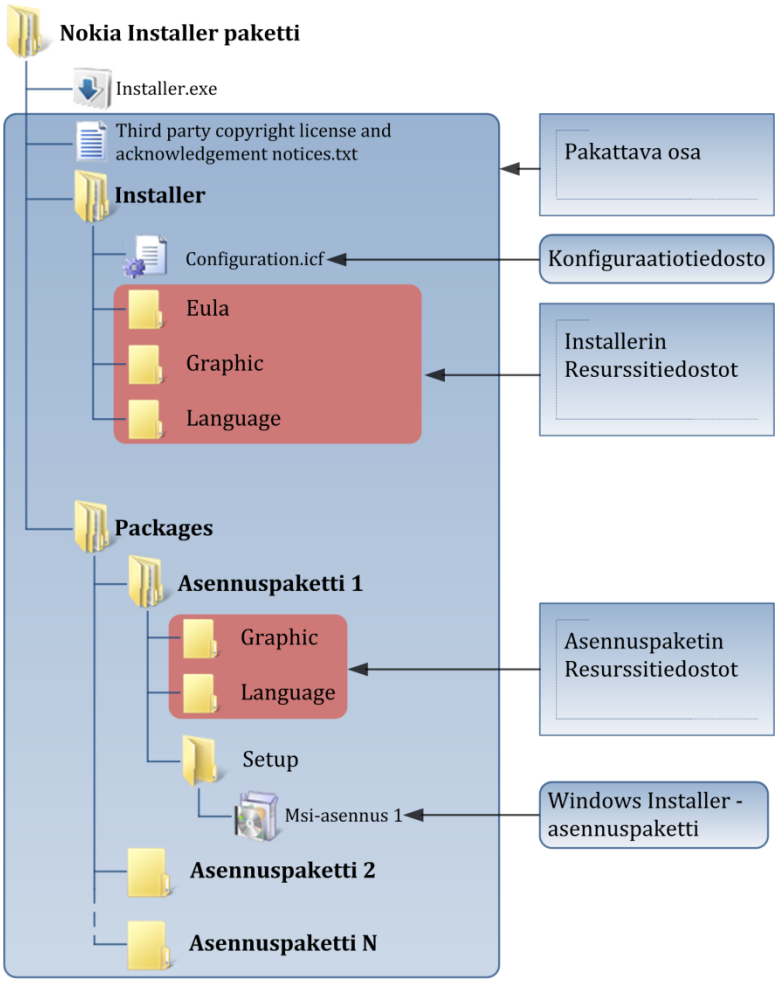
Nokia Installerin kehitys alkoi vuonna 2006 tarkoituksena ratkaista PC Suiten asennuksen kasvaneet vaatimukset. Sen ensisijaisena tehtävänä oli mahdollistaa useiden msi-asennusten ketjuttaminen ja asennusprosessin edistymisen välittäminen käyttäjälle. Lisäksi haluttiin käyttää tehokkaampaa tiedostopakkausta asennuspaketin koon pienentämiseksi ja uudistaa asennuksen käyttöliittymää ja ulkoasua. Teknologisesti Nokia Installer ei sisällä mitään uutta, vaan hyödyntää jo olemassa olevia teknologioita.

4.2 Pakkaus ja hakemistorakenne

Nokia Installer tukee sekä pakattua että pakkaamatonta asennusta. Pakattu asennuspaketti sisältää itsessään kaikki asennuksen tiedostot ja on tarkoitettu jaettavaksi Internetissä. Installer purkaa asennukseen tarvittavat tiedostot ajon aikana paketista paikalliselle kiintolevylle kuvion 20 mukaiseen hakemistorakenteeseen. Pakkaukseen käytetään LZMA-algoritmia, jolla saavutetaan zip-pakkausta parem-

pi pakkausaste. Jotta algoritmista saadaan kaikki hyöty irti, msi-pakettien CAB -pakkauksen pakkausaste on asetettu nollaksi, koska jo kertaalleen pakattu data ei pakkaudu uudelleen pakattaessa, vaan pahimmillaan sen koko kasvaa.

Pakkaamaton asennus on suunniteltu käytettäväksi CD-ROM levyltä. Silloin kuvion 20 mukainen hakemistorakenne on valmiina Nokia Installerin kanssa samassa hakemistossa.



KUVIO 20. Nokia Installer-paketin hakemistorakenne

Hakemistorakenteeseen kuuluu kaksi päähakemistoa: Installer ja Packages. Installer-hakemisto sisältää konfiguraatiotiedoston lisäksi hakemistot grafiikka- ja kielisekä lisenssitiedostoille. Packages-hakemisto sisältää asennuspaketit omissa hakemistoissaan. Asennuspaketeilla voi pakettityypistä riippuen olla omat grafiikka- ja kielitiedostonsa, jotka sisältävät asennuksen kuvakkeen, nimen ja kuvauksen.

Pakettityyppejä on neljä: pääpaketti, lisäpaketti, kolmannen osapuolen paketti ja piilotettu paketti.

Vain yksi asennuspaketti voi toimia pääpakettina. Nokia Installer valitsee sen asennustilan mukaan käynnistystilakseen joko uuden asennuksen tai asennetun tuotteen ylläpidon. Pääpaketin tuotenimeä käytetään koko asennuksen tuotenimenä, joten se pitää olla määritettynä paketin kielitiedostoissa.

Lisäpaketit ovat pääohjelmiston rinnalle asennettavia lisäsovelluksia, jotka ovat käyttäjän valittavissa, mikäli:

- Asennus on uusi ja pääpaketti on valittu asennettavaksi.
- Asennus on ylläpitotilassa, jolloin pääpaketti on asennettu jo aiemmin ja lisäpaketteja asennetaan jälkikäteen.
- Asennus on uusi ja pääpakettia ei ole valittu asennettavaksi, mutta siitä on asennettuna aiempi versio.

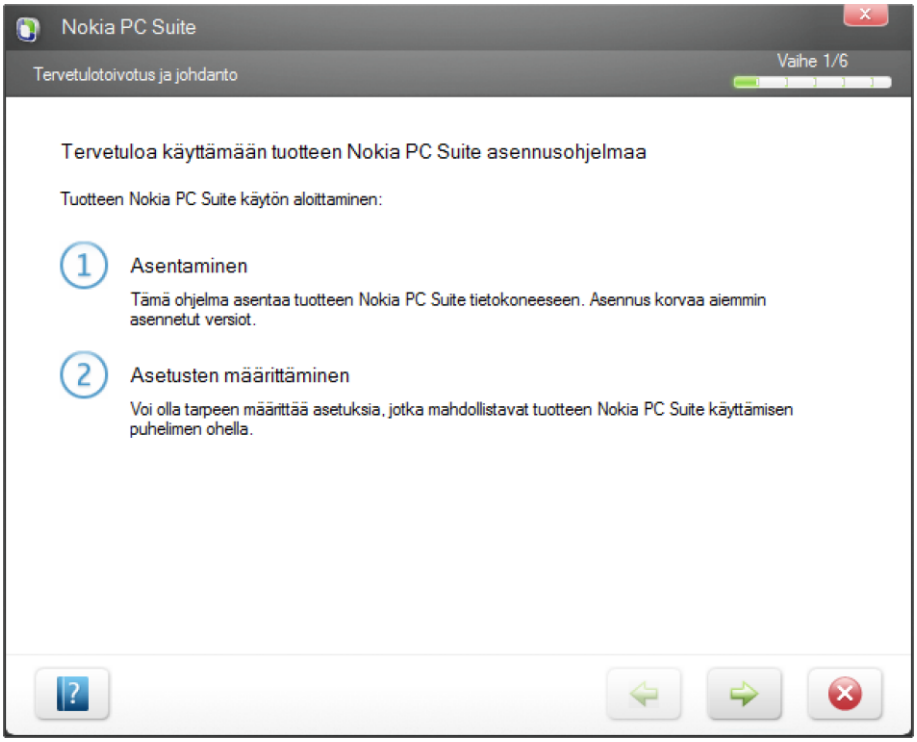
Nokia Installer ei poista lisäpaketteja tuotetta poistettaessa.

Kolmannen osapuolen paketit ovat pääohjelmistosta riippumattomia ohjelmistoja, joita voidaan asentaa vaikka pääpakettia ei ole valittu asennettavaksi. Nokia Installer ei voi asentaa kolmannen osapuolen ohjelmistoa ilman, että käyttäjä hyväksyy sen oman lisenssin, joten nämä ohjelmistot asennetaan käyttäen niiden omaa asennusta. Nokia Installer ainoastaan käynnistää asennuksen ja jää odottamaan sen päättymistä. Kolmannen osapuolen ohjelmiston asennusta ei ole sidottu Windows Installer -teknologiaan, joten paketit voivat olla msi- tai exe-tiedostoja. Nokia Installer ei poista kolmansien osapuolten ohjelmistoja tuotetta poistettaessa.

Paketit, jotka eivät kuulu edellä mainittuihin tyyppeihin, ovat piilotettuja paketteja. Niitä ei näytetä Installerin käyttöliittymässä, joten niille ei tarvitse luoda resurssitiedostoja.

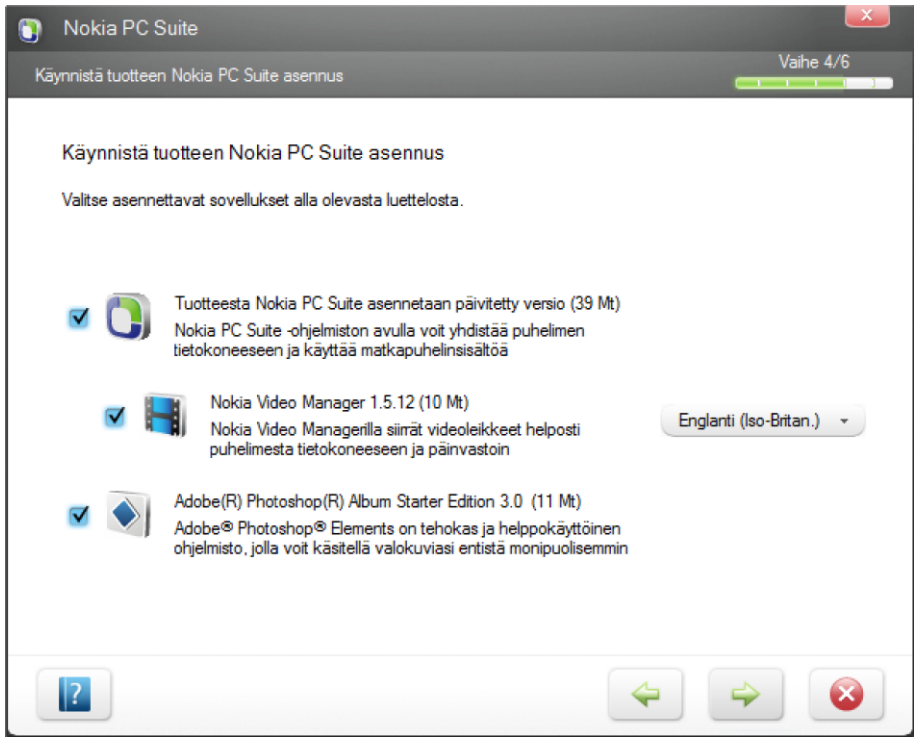
4.3 Käyttöliittymä

Yksi Nokia Installerin päätehtävistä on useiden Windows Installer -asennusten asentamisen hallitusti siten, että koko asennuksen edistyminen välittyy käyttäjälle. Tarkan edistymisen indikoimiseksi sekä yksittäisten asennusten viestien välittämiseksi Nokia Installer toimii Windows Installerin ulkoisena käyttöliittymäkäsittelijänä, joka korvaa msi-asennusten Windows Installer -tyylisen käyttöliittymän omallaan, kuvion 21 tyylisellä käyttöliittymällä.



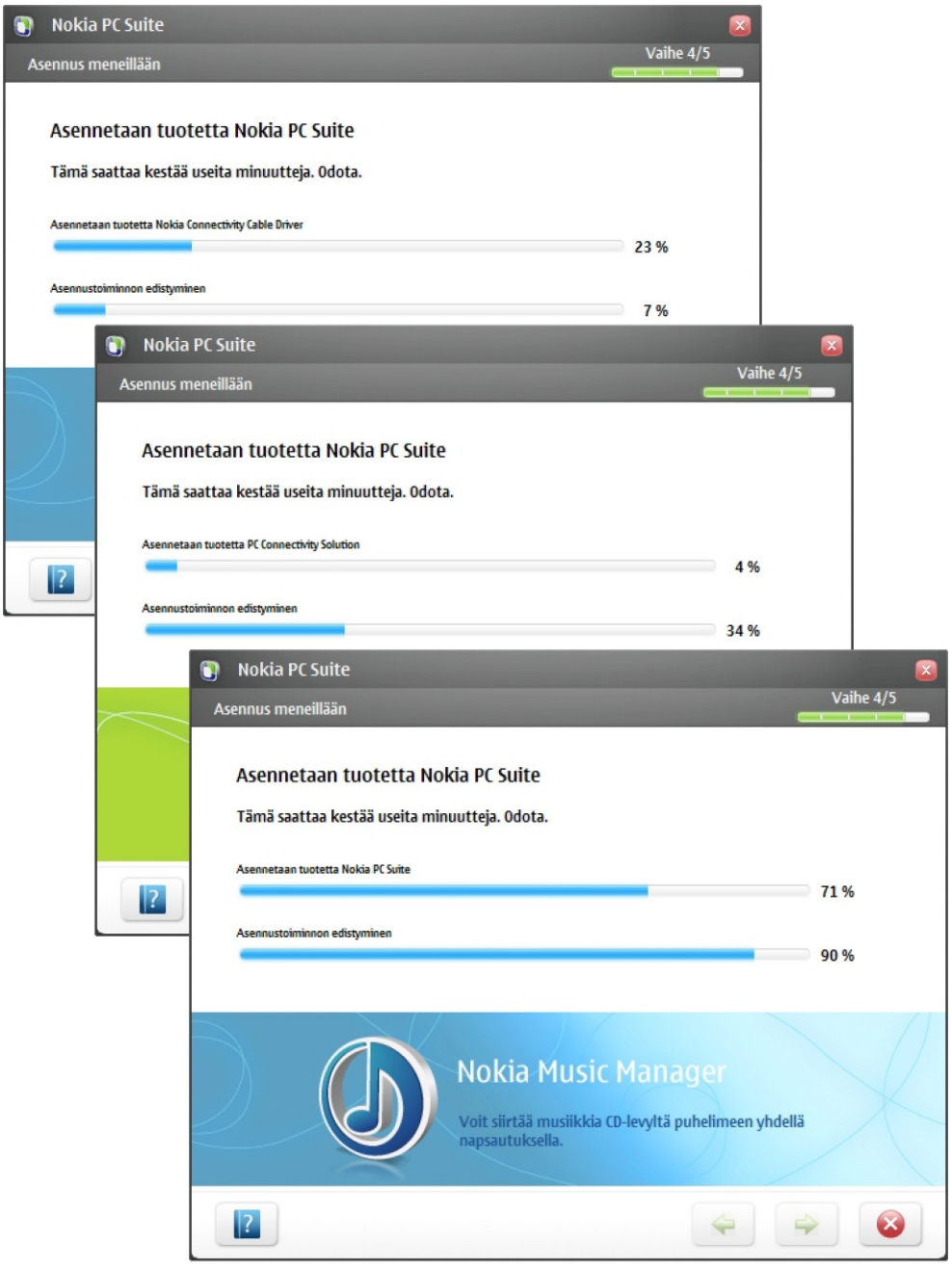
KUVIO 21. Nokia Installer

Perinteisissä asennuksissa käyttäjällä on mahdollisuus valita, mitä ominaisuuksia yksittäisestä asennuspaketista asennetaan. Nokia Installer poikkeaa tästä siten, että yksittäinen asennus asennetaan sen oletusominaisuuksilla ja käyttäjälle tarjottavat valintamahdollisuudet liittyvät kuvion 22 mukaisesti *Installeripakettiin* sisältyvien asennusten valintaan.



KUVIO 22. Nokia Installerin version 2.0 sovellusten valintadialogi

Kuviossa 23 on esitetty Nokia Installerin version 1.0 asennusdialogi, jossa käytettiin kahta edistymispalkkia, joista toinen kuvasi yksittäisen asennuksen edistymistä ja toinen koko asennusta. Verrattuna PC Suiten aiempaan asennukseen, jossa yhdestä msi-asennuspaketista asennettiin useita tuotteita, Nokia Installerin käyttäjälle välittämä informaatio oli selkeä parannus. Versiossa 2.0 yksittäisten asennusten edistymisilmaisinta ei enää käytetty, koska sen ei katsottu antavan käyttäjälle merkittävää lisäinformaatiota asennuksen kulusta. Lisäpakettien ja kolmansien osapuolten ohjelmistojen asennukset sen sijaan näytetään.



KUVIO 23. Nokia Installerin version 1.0 asennuksen edistymisdialogi

4.4 Konfigurointi

Nokia Installerin toiminta kuvataan konfigurointitiedostossa *configuration.icf*. Se on rakenteeltaan lohkoista ja avain-arvopareista koostuva tekstitiedosto, oheisen kuvion 24 mukaisesti.

```
//Kommentti
[LOHKO1]
Avain1=Arvo1
Avain2=Arvo2

[LOHKO2]
Avain1=Arvo1
Avain2=Arvo2
…
```

KUVIO 24. Nokia Installerin konfiguraatiotiedoston syntaksi

Konfiguraatio määrittää asennuspaketin yleiset ominaisuudet sekä pakettiin kuuluvat msi-asennukset. Yleisiä ominaisuuksia ovat:

- julkaisija
- esimääritelty asennushakemisto Windowsin ohjelmahakemistossa
- pääasennus
- lista valinnaisista lisäasennuksista
- lista valinnaisista kolmansien osapuolten ohjelmistojen asennuksista
- näytettävät asennusdialogit
- Windows Installerin lokitiedoston yksityiskohtaisuus
- lista käytettävistä kielistä
- oletuskieli
- asennuksen aikana näytettävien mainostaulujen näyttöaika
- fontti, fonttitiedoston nimi
- WWW-linkit ohjesivustolle ja FAQ-sivustolle.

Pakettien asennusjärjestys määräytyy niiden esiintymisjärjestyksen mukaan ylhäältä alaspäin luettaessa. Asennuspaketeille määritellään:

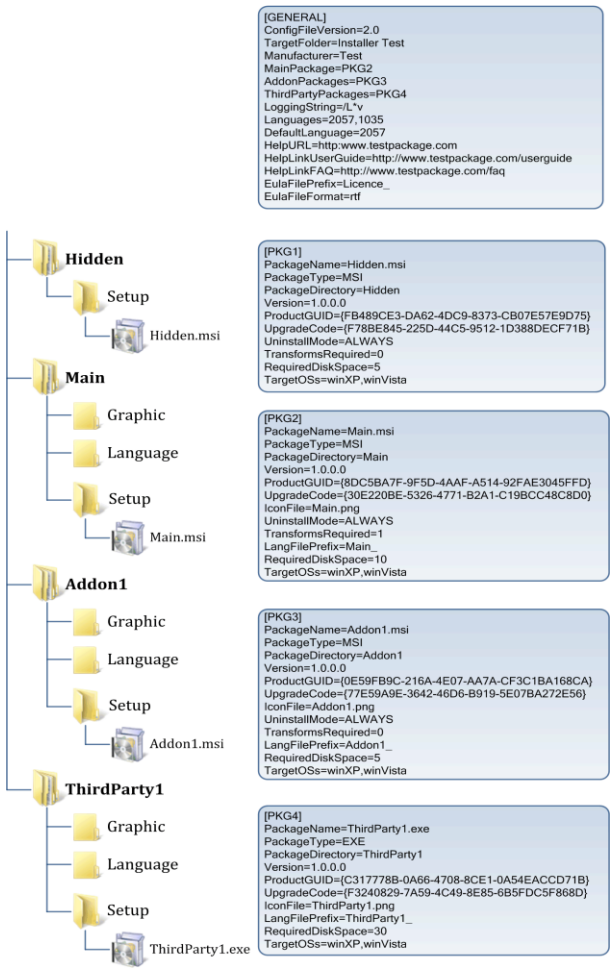
- msi-paketin tiedostonimi
- versio
- asennustyyppi, msi/exe
- asennuspaketin hakemistonimi
- tuotekoodi
- päivityskoodi
- kieliresurssitiedoston nimen etuliite
- kuvaketiedosto

- poistotapa
- käyttääkö asennus Windows Installerin tietokantamuunnoksia
- tilavaatimus.

Muita määriteltäviä ominaisuuksia ovat:

- riippuvuudet muista ohjelmistoista
- yhteensopivuudet muihin ohjelmistoihin
- päivitysten tarkistaminen Internetistä
- ohjelmien käynnistäminen asennuksen jälkeen
- asennuksen aikana näytettävät mainostaulut.

Kuviossa 25 on esitetty vierekkäin Installeripaketin konfiguraatio ja sitä vastaava hakemistorakenne.



KUVIO 25. Konfiguraatio ja sitä vastaava hakemistorakenne

4.5 Rajoitukset ja puutteet

Nokia Installeria käytettäessä siinä on havaittu olevan puutteita, jotka vaikuttavat siihen, millaisia asennuspaketteja sille on mahdollista luoda. Pakettityypit rajoittavat asennusjärjestystä, mm. Microsoftin valmiiden Visual C++ -jakelupakettien käyttäminen ei toistaiseksi ole mahdollista, koska ne ovat tiedostotyypiltään exe-tiedostoja, joita tuetaan vain kolmansien osapuolten paketeille, joita ei ole mahdollista asentaa ennen pääpakettia.

4.6 Aputyökalut

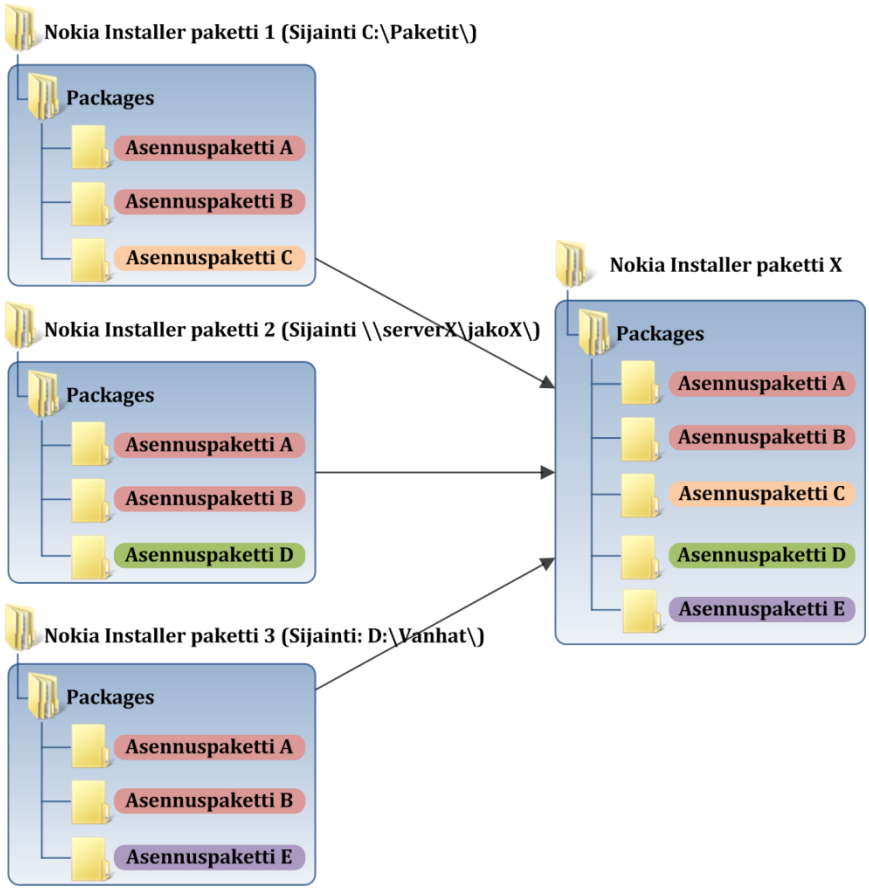
Nokia Installerin tueksi on tehty kaksi aputyökalua: konfiguraation rakennustyökalu sekä tarkistustyökalu. Tarkistustyökalu on komentorivipohjainen sovellus Installeripaketin hakemistorakenteen ja konfiguraatiotiedoston arvojen oikeellisuuden tarkistamiseen ja korjaamiseen. Sitä hyödynnetään konfiguraation rakennustyökalussa sekä asennuspakettien rakennusautomatiikassa.

Konfiguraatiotyökalu on graafinen sovellus, jolla voidaan luoda uusia asennuspaketteja yhdistelmällä eri Nokia Installer -pakettien sisältöjä. Työkalu on tämän opinnäytetyön toteutettava osuus, ja sitä käsitellään tarkemmin luvussa 5.

5 NOKIA INSTALLERIN KONFIGURAATIOTYÖKALU

5.1 Tausta

Nokia Installerin asennuspakettien määrittämisessä oli tarvetta sovellukselle, jonka avulla voidaan helposti luoda uusia asennuskonfiguraatioita ilman tarkempaa tuntemusta installerin konfiguroinnista tai sen sisäisestä hakemistorakenteesta. Työkalulla ei ollut tarkoitus luoda konfiguraatioita alusta asti, jolloin käyttäjän olisi pitänyt määrittää kaikki konfiguraation asetukset, vaan kuvion 26 mukaisesti yhdistellä olemassa olevia installeripaketteja halutunlaiseksi kokonaisuudeksi.



KUVIO 26. Konfiguraatiotyökalun käyttötarkoitus

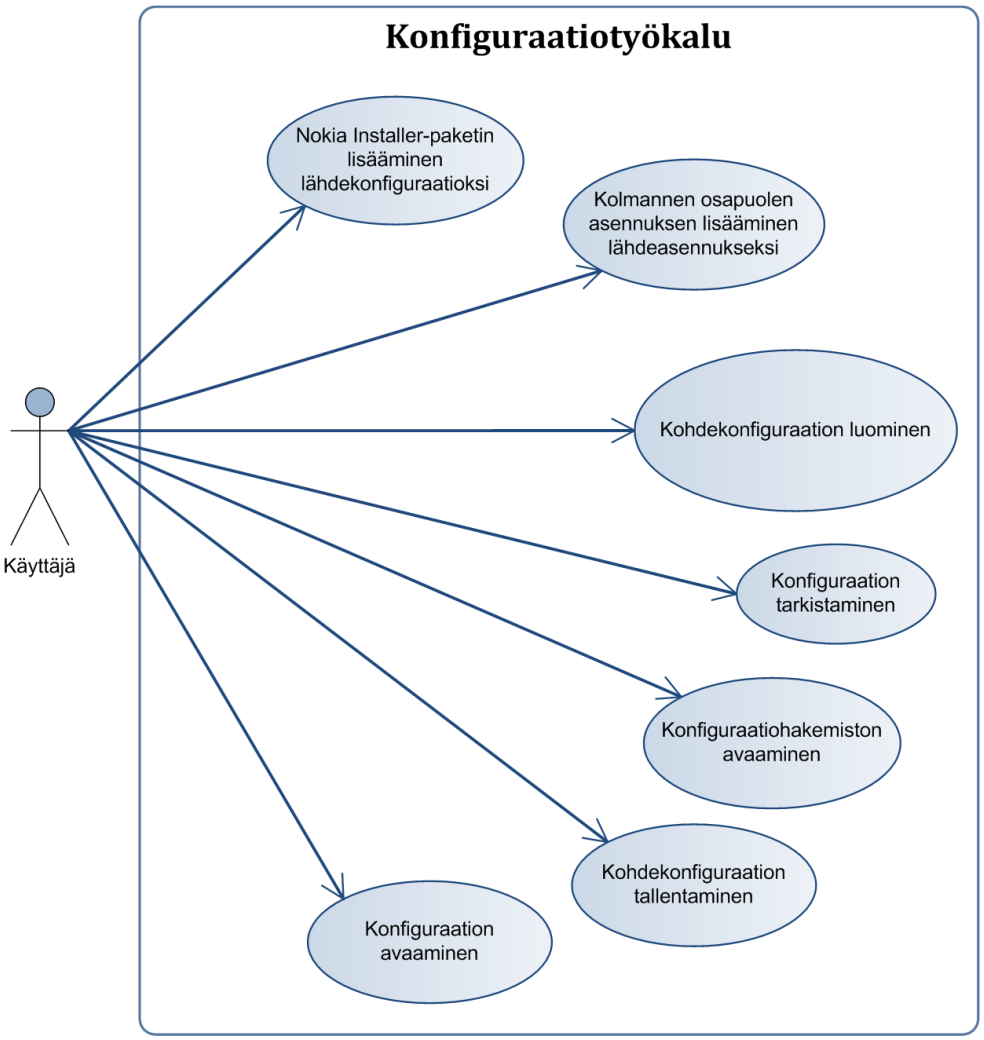
5.2 Vaatimukset

Projektin esittelyssä sovellukselle laadittujen toiminnallisten vaatimusten mukaan käyttäjä voi:

- lisätä sovellukseen valmiita Nokia Installer -paketteja käytettäväksi lähdekonfiguraatioina
- luoda konfiguraatiokuvauksia. Konfiguraatiokuvaus sisältää tiedot Installerin konfiguraatiotiedoston arvoista sekä konfiguraatioon liitettävien tiedostojen sijainnit. Sovellus rakentaa lopullisen konfiguraation eli Nokia Installeripaketin konfiguraatiokuvauksen perusteella
- nimetä konfiguraatiokuvauksen
- tallentaa konfiguraatiokuvauksen tiedostoon
- avata tallennetun konfiguraatiokuvauksen
- muokata konfiguraatiokuvausta: muuttaa konfiguraatiotiedoston arvoja ja määrittää konfiguraatioon rakennettavat komponentit (asennuspaketit, mainostaulut, ym.)
- määrittää kohdehakemiston rakennettavalle konfiguraatiolle
- rakentaa pakkaamattoman tai pakatun Nokia Installer -paketin konfiguraatiokuvauksesta
- avata rakennetun paketin hakemiston sovelluksesta käsin (aukaisee uuden Windows Explorer ikkunan)
- luoda uusia komponentteja konfiguraatiokuvaukseen, esimerkiksi lisätä asennuksen, jota ei ole mukana aiemmin tehdyissä Installer -paketeissa
- tallentaa komponentin kuvauksen tiedostoon.
- lisätä aiemmin talletetun komponentin kuvauksen konfiguraatiokuvaukseen.
- poistaa komponentin konfiguraatiokuvauksesta.

Käyttäjän suorittaessa toimintoja sovellus raportoi tapahtumista käyttöliittymän tulosikkunaan. Lisäksi sovellus hyödyntää konfiguraation tarkistustyökalua rakennettujen konfiguraatioiden oikeellisuuden varmistamiseen.

Koska kyseessä oli sisäiseen käyttöön tarkoitettu erillinen työkaluohjelma, toiminnallista määrittelyä ei tehty edellä mainittua tarkemmaksi, eikä käyttötapauksista laadittu yksityiskohtaisia kuvauksia. Keskeisimmät käyttötapaukset on koottu kuvion 27 käyttötapauskaavioon.



KUVIO 27. Konfiguraatiotyökalun tärkeimmät käyttötapaukset

5.3 Kehitys- ja käyttöympäristöt

Konfiguraatiotyökalun ohjelmoinnissa olisi ollut mahdollista hyödyntää Nokia Installerin ohjelmakoodia. Installer oli kuitenkin toteutettu Win32 -sovelluksena, josta ainoa hyödynnytettävissä oleva osa olisi ollut konfiguraatiotiedoston lukemiseen liittyvät toiminnot. Käyttöliittymä ja muut toiminnot oli joka tapauksessa tehtävä alusta alkaen, joten työkalun toteuttaminen .NET 2.0 -pohjaisena Windows Forms -sovelluksena tuntui Win32:ta tai MFC:ia houkutteluvammalta vaihtoehdolta, koska .NET -kirjastojen käyttö vähentäisi huomattavasti itse toteutettavan ohjelmakoodin määrää ja siten vähentäisi ohjelmointivirheitä. Ohjelmointikieleksi valittiin C#, johon vaikutti ajatus käyttää ohjelman käyttöliittymässä konfiguraation ominaisuuksien esittämiseen www.CodeProject.com:ssa julkaistua *PropertyGridEx* -käyttöliittymäkomponenttia.

Kehitysympäristö:

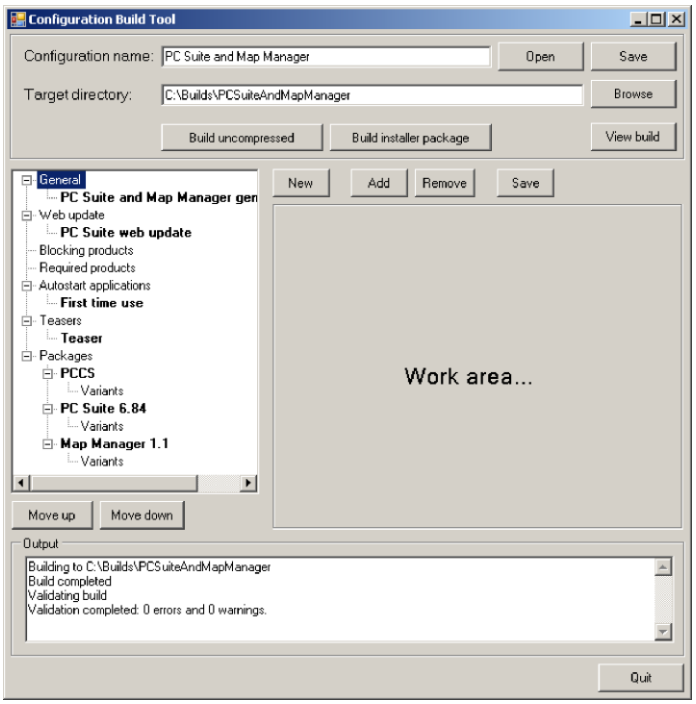
- Visual Studio 2008
- C#
- Windows Forms -sovellus
- .NET 2.0

Sovellus oli tarkoitettu käytettäväksi seuraavissa 32- ja 64-bittisissä Windows -käyttöjärjestelmissä:

- Windows XP (Service Pack 2)
- Windows Vista.

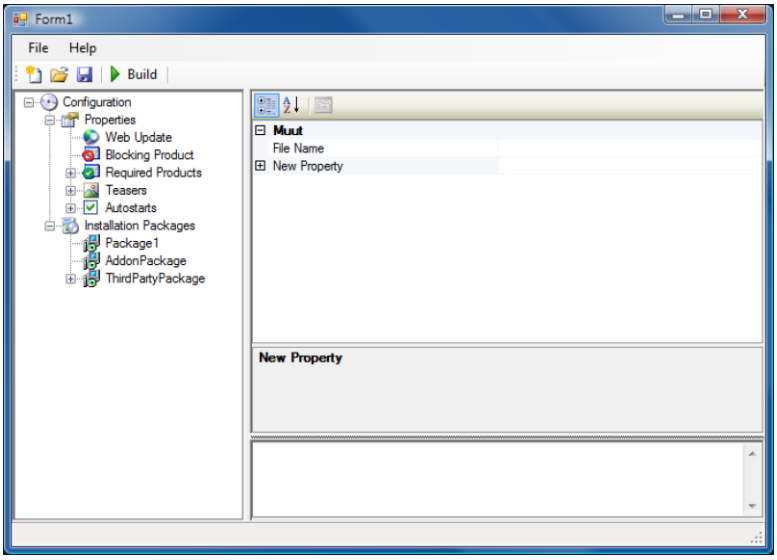
5.4 Käyttöliittymäprototyypit

Määrittelydokumenttien sijasta suunnittelussa edettiin käyttöliittymäprototyypeillä, joista ensimmäisenä voidaan pitää projektin esittelyssä toimintojen esittämiseen käytettyä kuvion 28 käyttöliittymäesimerkkiä.



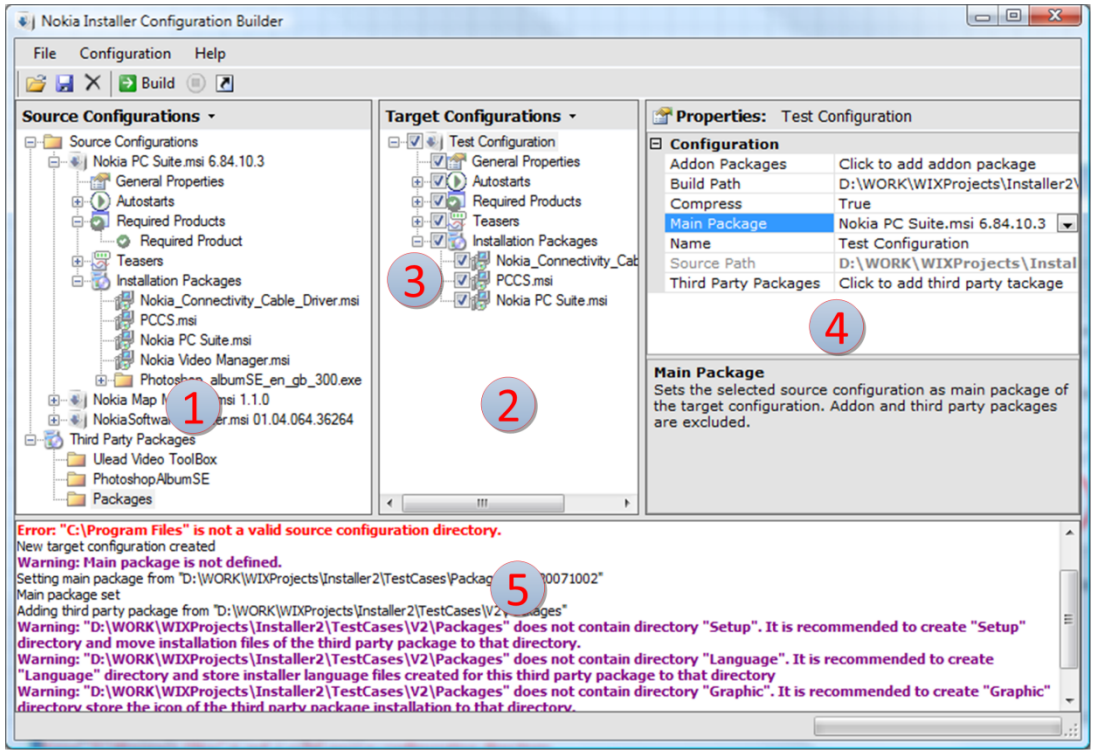
KUVIO 28. Ensimmäinen käyttöliittymäprototyyppi

Ensimmäistä prototyyppiä mukaillen toteutettiin kuvion 29 mukainen toinen prototyyppi, jossa käytettiin *PropertyGridEx* -komponenttia valitun kohteen ominaisuuksien näyttämiseen ja muokkaamiseen. Tässä vaiheessa toiminnallisuutta ei vielä ollut toteutettu ja sovelluksen toimintaa simuloitiin edelleen ajatustasolla.



KUVIO 29. Toinen käyttöliittymäprototyyppi

Toinen prototyyppi osoittautui riittämättömäksi määriteltyjen toimintojen toteuttamiseksi, joten kolmannessa prototyypissä käyttöliittymä laajennettiin kuvion 30 mukaiseksi. Lähde- ja kohdekonfiguraatiot sijoitettiin omiin kehysnäkymiinsä, joissa konfiguraatiot esitettiin edelleen puurakenteina (kuvioon 30 merkityt *kohdat 1 ja 2*). Kohdekonfiguraatiopuun elementteihin lisättiin valintaruudut (*kohta 3*), joilla konfiguraation sisältöä voidaan muuttaa poistamatta elementtejä pysyvästi konfiguraatiosta. Vain valitut osat rakennetaan konfiguraatioon. Ominaisuusnäkymään (*kohta 4*) ei tehty muutoksia. Tulosnäkymä (*kohta 5*) siirrettiin omaan, koko ikkunan levyiseen kehykseensä. Kolmas prototyyppi hyväksyttiin sovelluksen käyttöliittymäksi.

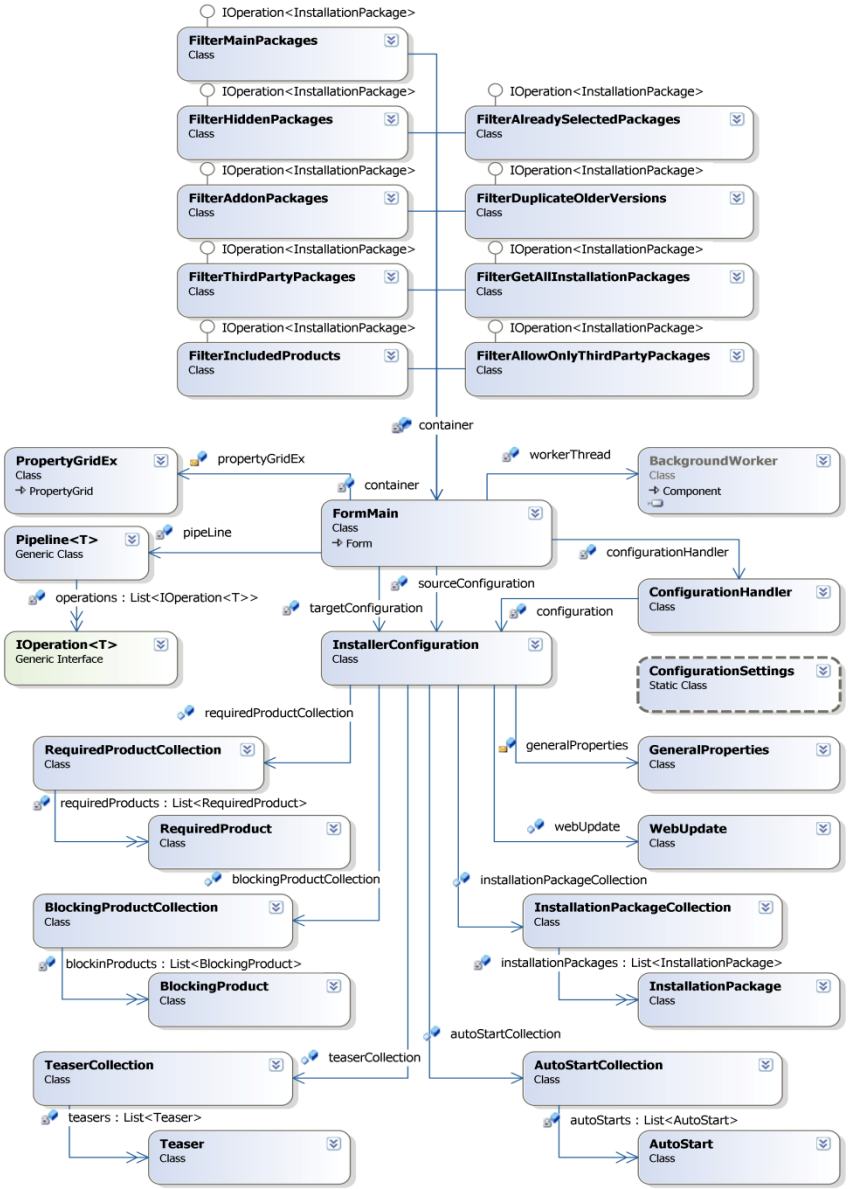


KUVIO 30. Kolmas käyttöliittymäprototyyppi

5.5 Oliomalli

5.5.1 Luokkakaavio

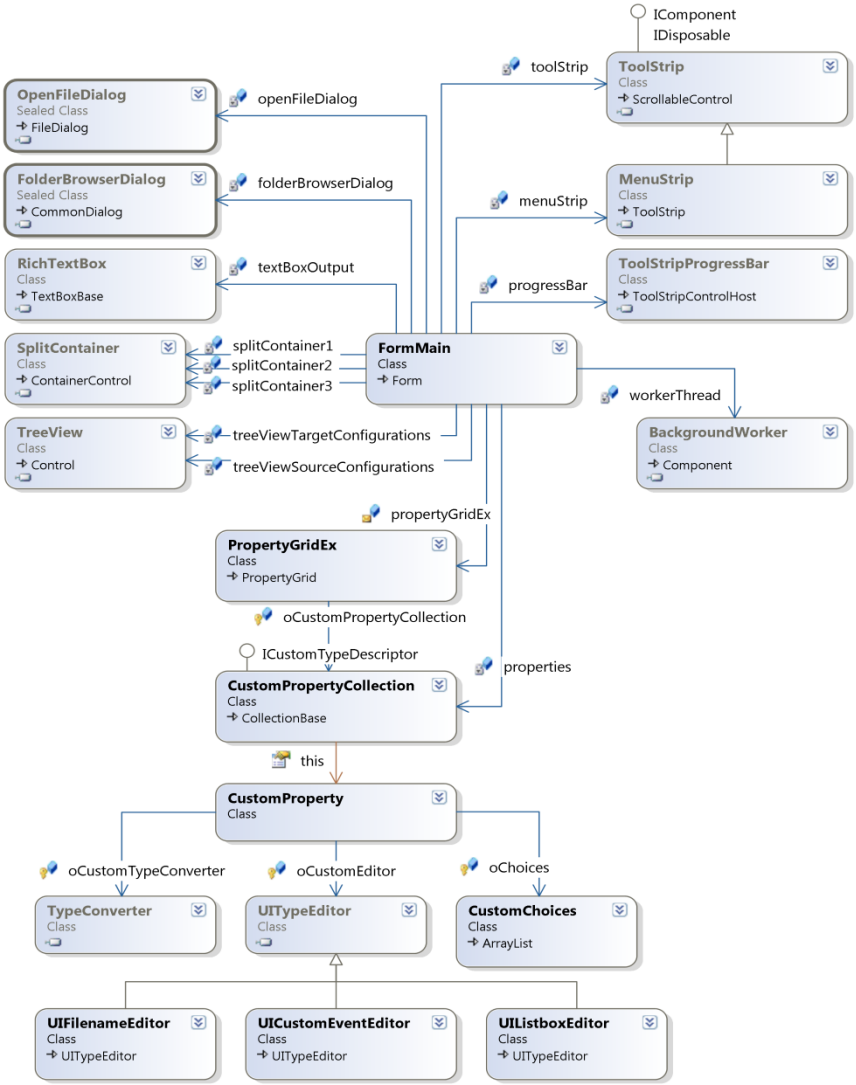
Konfiguraatiotyökalun toteuttavat luokat on esitetty kuvion 31 luokkakaaviossa. Visual Studion dialogieditorin lisäämiä käyttöliittymäkomponentteja ei ole esitetty kaaviossa, vaan ne sisältyvät *FormMain* -luokkaan. *FormMain* sekä *PropertyGridEx* vastaavat tietojen esittämisestä käyttöliittymässä, muut luokat liittyvät konfiguraatiotiedon käsittelyyn. Kaavion luokkia tarkastellaan lisää jäljempänä.



KUVIO 31. Luokkakaavio

5.5.2 Käyttöliittymäluokat

Konfiguraatiotyökalun käyttöliittymään liittyvät luokat on esitetty tarkemmin kuviossa 32. *PropertyGridEx* -komponenttia lukuun ottamatta kaikki ohjelmassa käytetyt käyttöliittymäkomponentit kuuluvat Visual Studio 2008:n vakiona tarjoamiin komponentteihin. *FormMain* on käyttöliittymän pääluokka, johon komponentit on liitetty. Ohjelmaikkunan eri osien koon on muuttamiseksi *FormMain* sisältää kolme *SplitContainer* -komponenttia, jonka käyttötarkoituksena on jakaa sisältämänsä alue vaaka- tai pystysuorassa kahteen osaan, osien väliin jäädessä siirreltävä erotinpalkki. Kaksi *Treeview* -komponenttia on puunäkymäkomponentit konfiguraatioiden elementtien esittämiseen.



KUVIO 32. Käyttöliittymäluokat

Valitun elementin ominaisuudet näytetään *PropertyGridEx* -komponentissa taulukkomuodossa, jonka yksittäiset solut esitetään joko:

- sellaisenaan merkkijonona
- yksinkertaisena valintalistana
- tiedosto- tai hakemistopolkuna, jolloin solun valitseminen tuo esille painikkeen, josta avautuu tiedoston tai hakemiston avausdialogi ominaisuuden tyypin mukaisesti. Solun arvoa voidaan myös muokata suoraan.

PropertyGridEx:n ominaisuudet ovat käytettävissä vain, jos komponentilla näytetään *CustomProperty* -tyyppisiä olioita. Oliot säilötään *CustomPropertyCollection* -olioon. *PropertyGridEx*:n ”SelectedObject” -ominaisuuteen asetetaan kulloinkin näytettävän olion *CustomPropertyCollection* -olio, jolloin *PropertyGridEx* näyttää kokoelmaolioon talletetuttujen *CustomProperty* -olioiden ominaisuudet.

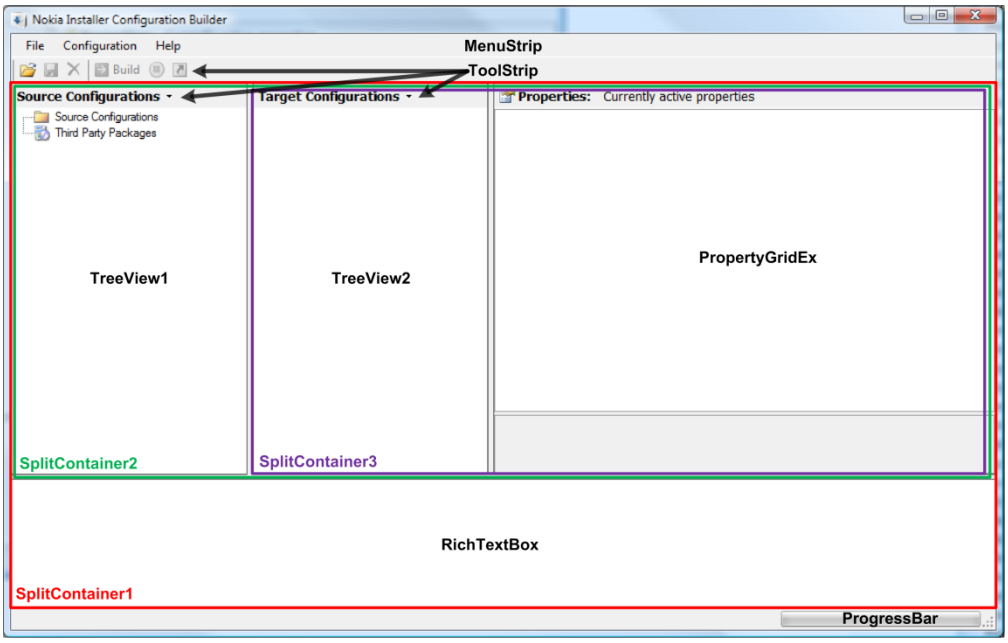
RichTextBox -olio toimii ohjelman tulosikkunana, johon tulostetaan tietoja toimintojen tuloksista, virheilmoitukset ja varoitukset sekä ohjeita käyttäjälle. Tulosikkunaan kirjoittaminen muista säikeistä toteutettiin delegaatilla, joka on C#:n vastine C++:n funktio-osoittimelle.

BackgroundWorker -komponentti suorittaa sille määritetyn tehtävän omassa säikeessään, jolloin käyttöliittymä säilyy vastaanottavana aikaa vievän toiminnon suorituksessa. *BackgroundWorkeria* käytetään konfiguraation rakentamisessa, joka on ohjelman raskain operaatio. Pitkäkestoisen operaation edistymisestä ilmoitetaan käyttäjälle *ToolStripProgressBar* -komponentin välityksellä.

OpenFileDialog ja *FolderBrowseDialog* -komponentteja käytetään lähdekonfiguraatioiden ja kolmansien osapuolten asennusten lisäämisessä valikkotoimintojen kautta tehtynä sekä tallennetun konfiguraation avaamisessa.

ToolStrip ja *MenuStrip* -komponentteja käytetään ohjelman päävalikon ja työkalurivin toteuttamiseen.

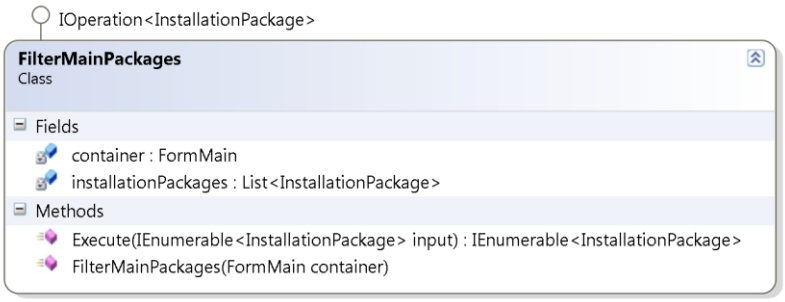
Kuviossa 33 on esitetty komponenttien asemoinnit. *SplitContainer1* jakaa ohjelmaikkunan vaakasuorassa kahtia, ylemmän osan sisältäessä *TreeView- ja PropertyGridEx* -komponentit ja alemman *RichTextBox* -komponentin. *SplitContainer2* jakaa *SplitContainer1*:n ylemmän osa kahtia pystysuorassa siten, että *TreeView1* jää vasempaan osaan ja *TreeView2* sekä *PropertyGridEx* oikeaan. *SplitContainer3* jakaa *TreeView2*:n ja *PropertyGridEx*:n omiin osioihinsa.



KUVIO 33. Komponenttien asemointi

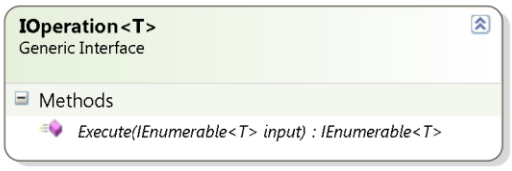
5.5.3 Pipeline ja Filter-luokat sekä IOperation -rajapinta

Näytettäessä listoja lähdekonfiguraatioista kohdekonfiguraatioon siirrettävissä olevista asennuspaketeista, ohjelman on suodatettava listoilta väärän typpiset asennukset, jo valitun asennuksen kopiot ja vanhemmat versiot. Suodatustoiminnallisuus toteutettiin käyttäen ”pipes and filters” -mallia, jossa lähdetietoa suodatetaan erilaisilla suodattimilla, kunnes jäljellä on vain haluttu tieto. Sivulla 53 esitetyn kuvion 31 luokkakaaviossa Filter-alkuiset luokat ovat ohjelmassa käytettyjä suodattimia, joiden toteutus on kuvion 34 kaltainen.



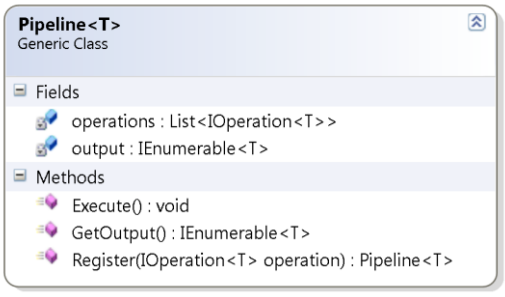
KUVIO 34. Filter -luokkien toteutus

Jokainen suodatinluokka toteuttaa kuvion 35 *IOperation* -rajapinnan *Execute* -metodin.



KUVIO 35. IOperation -rajapinta

Suodattimet rekisteröidään kuvion 36 *Pipeline*-luokan olioon ja lopuksi kutsutaan sen Execute-metodia. Suodatettu tieto on sen jälkeen luettavissa *GetOutput* -metodilla.



KUVIO 36. Pipeline-luokka

Suodatuksen koodiesimerkki on esitetty kuviossa 35.

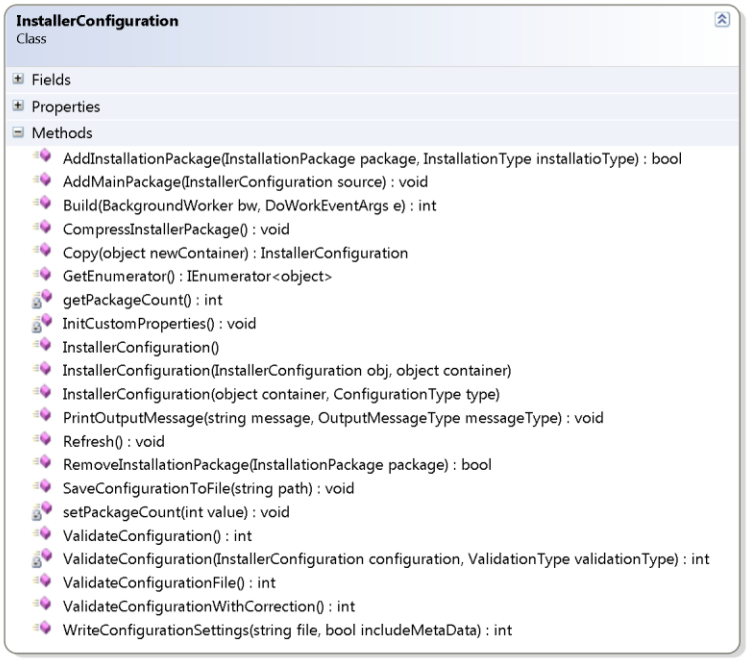
```
1  pipeLine = new Pipeline<InstallationPackage>();
2  pipeLine.Register(new FilterGetAllInstallationPackages(this));
3  pipeLine.Register(new FilterAlreadySelectedPackages(this));
4  pipeLine.Register(new FilterThirdPartyPackages(this));
5  pipeLine.Register(new FilterIncludedProducts(this));
6  pipeLine.Execute();
7  InstallationPackageCollection filteredAddonPackages;
8  filteredAddonPackages.InstallationPackages=
9          (List<InstallationPackage>)pipeLine.GetOutput();
```

KUVIO 37. Suodatusesimerkki

Rivillä yksi luodaan *PipeLine* -olion ilmentymä, jonka *List<IOperation<T>>* -tyyppiseen jäsenmuuttujaan suodattimet lisätään rivien 2–5 rekisteröinneissä. Ensimmäisenä rekisteröitävä suodatin toimii lähdetietona, josta muut suodattimet poistavat suodatettavia tietoja rivin 6 *Execute*-metodia suoritettaessa. Suodatettu tieto luetaan *Pipeline* -olion *GetOutput*-metodilla.

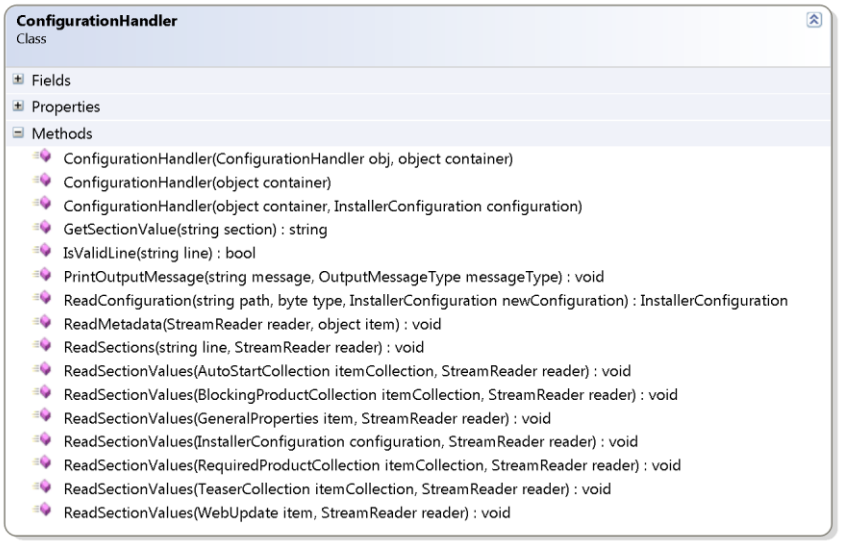
5.5.4 Konfiguraation käsittelyluokat

InstallerConfiguration -luokka sisältää yhteen konfiguraatioon liittyvät tiedot ja metatiedot. Siihen liittyvät muut luokat on esitetty kuvion 31 luokkakaaviossa sivulla 53 ja sen metodit kuviossa 38.



KUVIO 38. InstallerConfiguration -luokan metodit

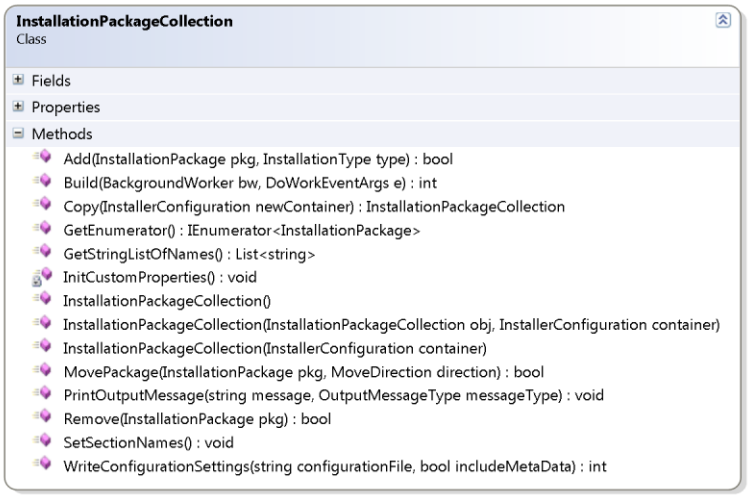
Hieman harhaanjohtavasti nimetyn *ConfigurationHandler* -olion ainoa tehtävä on konfiguraatiotiedoston lukeminen. Sen sisältämät, kuviossa 39 esitetyt metodit olisi luultavasti voitu siirtää *InstallerConfiguration* -luokkaan, jolloin *ConfigurationHandler* -luokkaa ei olisi tarvittu lainkaan. Näin ei kuitenkaan tehty.



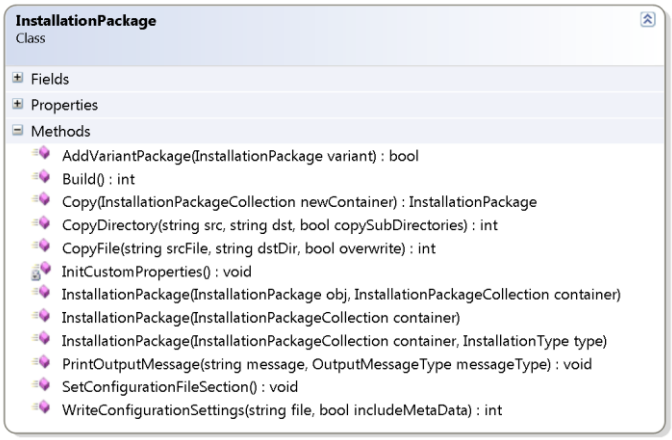
KUVIO 39. ConfigurationHandler -luokan metodit

ConfigurationSettings on staattinen luokka, joka sisältää ainoastaan vakiomäärittelyjä sekä luettelotietotyyppejä.

Kuvion 31 luokkakaaviossa sivulla 53 *InstallerConfiguration* -luokkaan liittyvät Collection-luokat toimivat säiliöinä ominaisuuksille, joita konfiguraatiossa voi olla mukana useampi kuin yksi esiintymä. Kukin ominaisuusluokka sisältää konfiguraatiotiedoston tietyn lohkon ominaisuudet. Kuvioissa 40 ja 41 on esitetty *InstallationPackage* -ominaisuuden säiliö- ja ominaisuusluokkien metodit. Säiliön metodeilla voidaan lisätä ja poistaa ominaisuuksia sekä vaihtaa niiden keskinäistä järjestystä konfiguraatiossa.

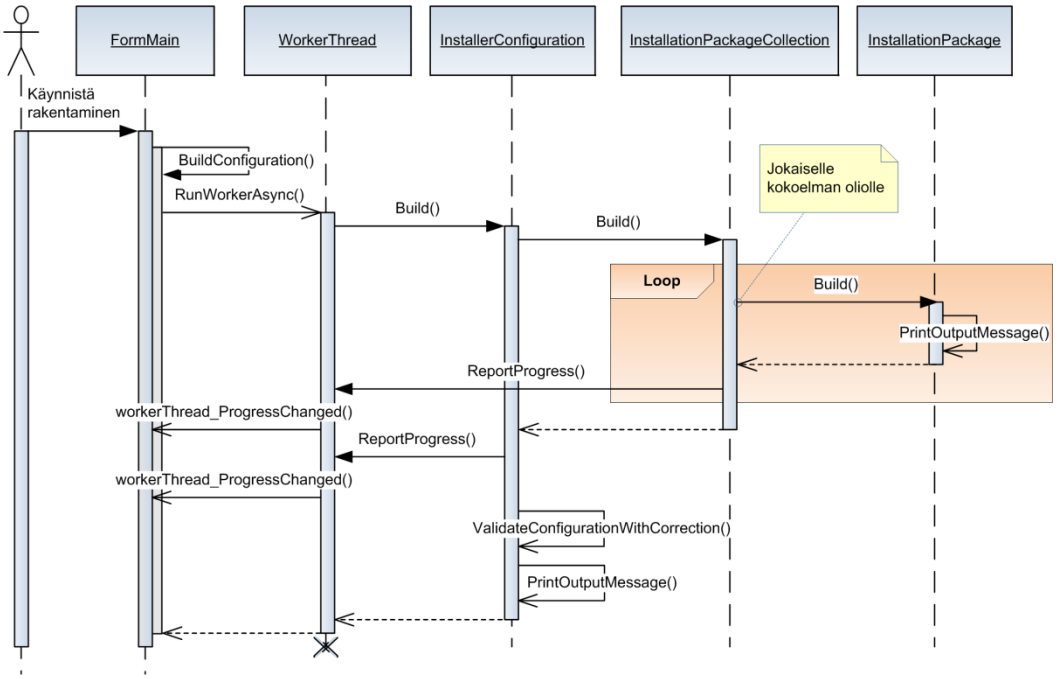


KUVIO 40. InstallationPackageCollection -luokan metodit



KUVIO 41. InstallationPackage -luokan metodit

Kuvion 42 sekvenssikaavion mukaisesti rakennettaessa konfiguraatiota *Installer-Configuration* -olio kutsuu kokoelman *Build*-metodia, jolloin kokoelmaolio kutsuu kunkin kokoelmaan kuuluvan ominaisuusolion *Build*-metodia. Muilla kokoelma–ominaisuuspareilla on vastaavat metodit ja niiden rakentaminen noudattaa samaa mekanismia. Konfiguraation rakentamisen valmistuttua konfiguraatio tarkistetaan tarkistustyökalulla, joka raportoi löydetyt virheet ja varoitukset tulosikkunaan.

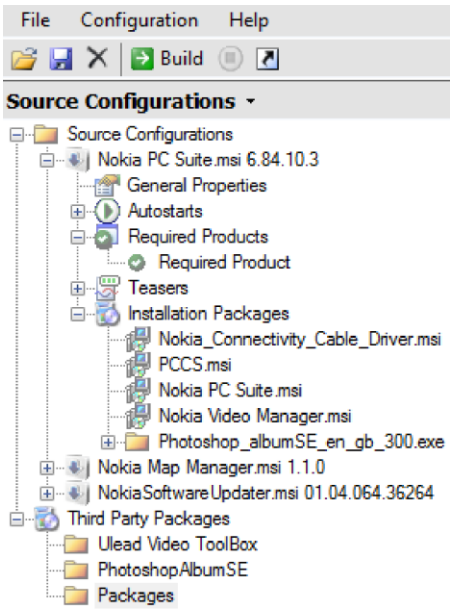


KUVIO 42. Konfiguraation rakentamisen sekvenssikaavio

5.6 Toiminnot

5.6.1 Lähdekonfiguraatioiden lisääminen

Lähdekonfiguraatiot lisätään kuviossa 43 esitettyyn lähdekonfiguraatio-osaan joko raahaamalla Installeripaketin juurihakemisto hiirellä ”Source Configurations” -haaraan tai valitsemalla lisäystoiminto ohjelman valikoista.



KUVIO 43. Lähdekonfiguraatio-osa

Raahaamalla voidaan lisätä myös pakattuja installeripaketteja, jolloin konfiguraatiotyökalu purkaa paketin ennen sen lisäämistä. Valikkotoiminnot toimivat ainoastaan pakkaamattomilla installeripaketeilla, koska niistä avautuu hakemistojen selausdialogi, jossa ei voi valita tiedostoja. Installeripakettia lisättäessä konfiguraatiotyökalu lukee paketin konfiguraatiotiedoston ja muodostaa sen avulla kuvion 43 mukaisen puurakenteen, jonka se nimeää paketin pääasennuksen mukaan. Konfiguraatiotiedoston asetusten lisäksi talletetaan lähdekonfiguraation sijainti tiedostojärjestelmässä. Koska installeripaketin hakemistorakenne on tunnettu, muut rakenteet hylätään ja epäonnistuneesta lisäyksestä tulostetaan virheilmoitus tulosnäkymään.

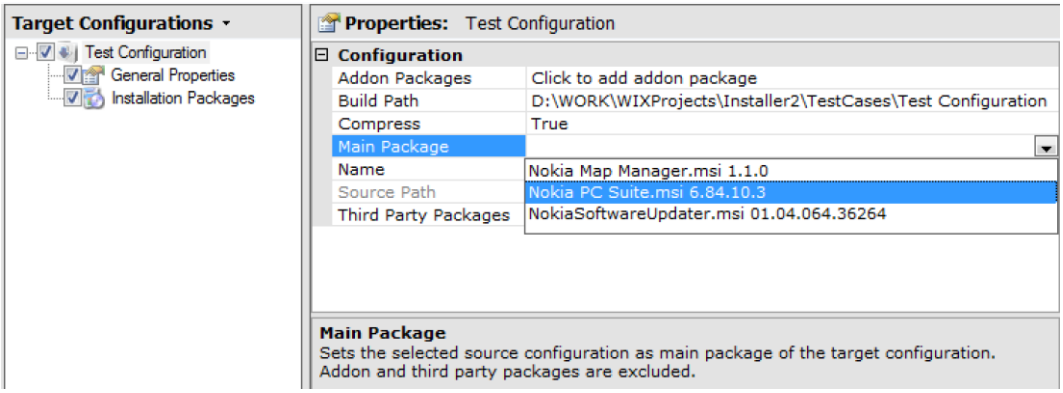
5.6.2 Kolmannen osapuolen asennuspaketien lisääminen

Erilliset kolmannen osapuolen asennukset lisätään lähdekonfiguraatio-osan ”Third Party Packages” -haaraan vastaavalla tavalla kuin pakkaamattomat installeripaketit. Koska erillisasennukset eivät sisällä installerin konfiguraatiotiedostoa, eikä niiden hakemistorakenne todennäköisesti sovellu sellaisenaan lisättäväksi Nokia Installeripakettiin, työkalu tarkistaa hakemistorakenteen ja ilmoittaa puutteista tulosikkunassa. Käyttäjän tehtäväksi jää hakemistorakenteen täydentäminen, ku-

vaketiedoston sekä asennuspaketin nimen ja kuvauksen sisältävän tekstitiedoston lisääminen.

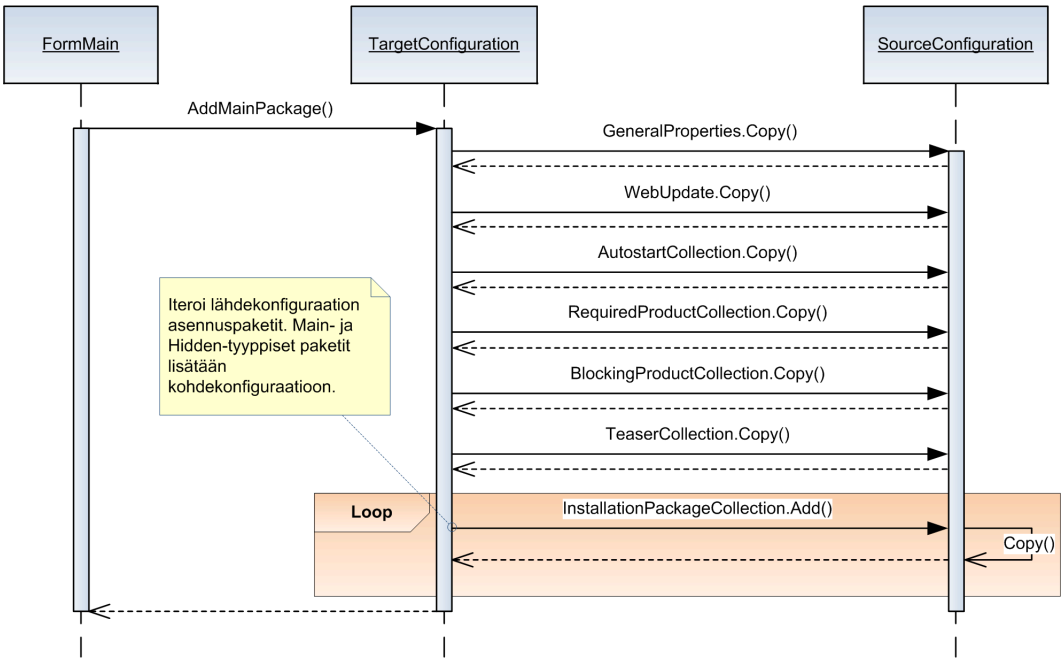
5.6.3 Uuden konfiguraation luominen

Uusi kohdekonfiguraatio luodaan joko valitsemalla ”File”-valikosta ”New Target Configuration” tai ”Target Configurations” -osan pudotusvalikosta ”New Configuration”, jolloin ohjelma luo tyhjän konfiguraation kohdekonfiguraatiopuuhun. Tämän jälkeen konfiguraation sisältö on valittavissa. Tyhjään konfiguraatioon on ensin valittava pääasennus kuvion 44 mukaisesti. Ohjelma lisää tällöin valitun asennuksen sekä sen piilotetut asennukset uuteen konfiguraatioon.



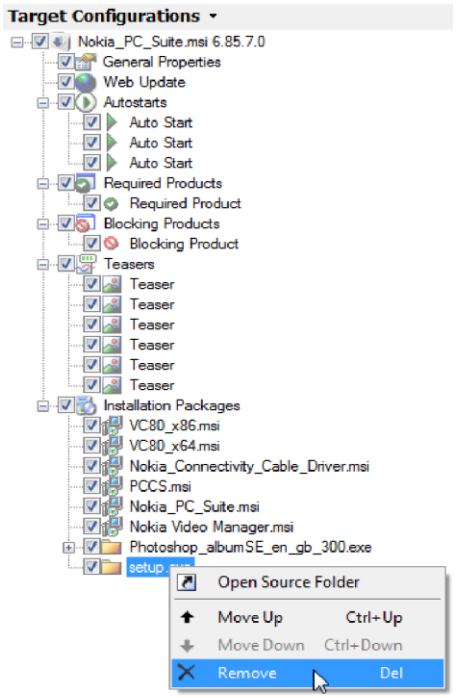
KUVIO 44. Pääasennuksen lisääminen kohdekonfiguraatioon

Ohjelmakoodissa pääasennuksen lisääminen tapahtuu kuvion 45 sekvenssin esittämällä tavalla. Kohdekonfiguraatio kutsuu lähdekonfiguraation ominaisuuksien *Copy*-metodia, joka kopioi ominaisuuden kohdekonfiguraatioon.



KUVIO 45. Pääasennuksen lisääminen kohdekonfiguraatioon

Pääasennuksen lisäämisen jälkeen konfiguraatioon voidaan valita lisäasennuksia ja kolmannen osapuolten asennuksia samaan tapaan kuin pääasennus valittiin kuviossa 44.

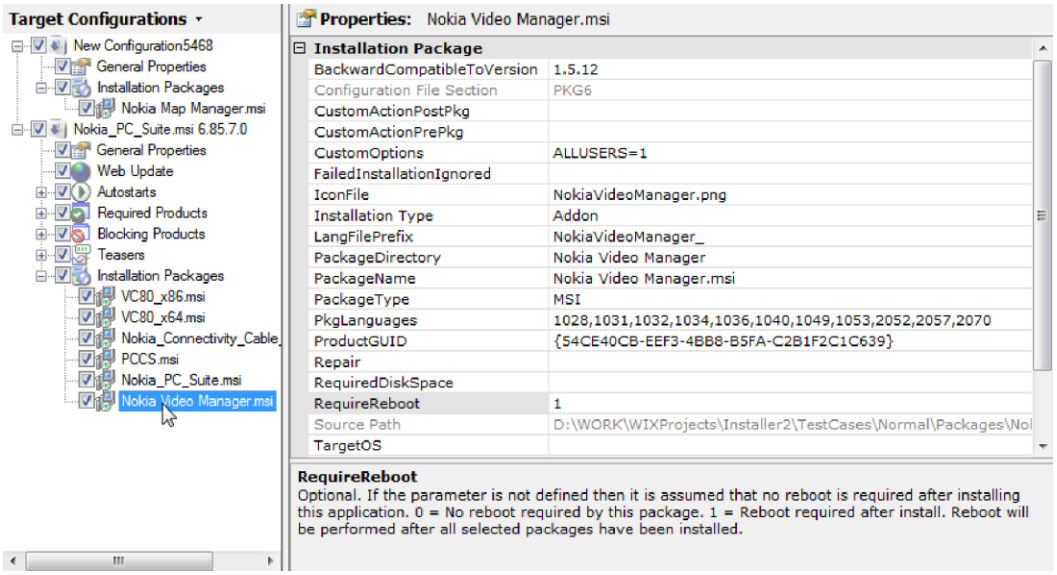


KUVIO 46. Kohdekonfiguraatiopuu

Kuvion 46 mukaisesti kohdekonfiguraation elementit on varustettu valintaruudulla, jolla elementti voidaan poistaa käytöstä. Pysyvästi poistettavia ovat ainoastaan lisäasennukset ja kolmansien osapuolten asennukset, jotka poistetaan joko delete -näppäimellä tai valitsemalla asennuspaketille aukeavasta ponnahdusvalikosta komento ”Remove”.

Asennuspakettien keskinäinen asennusjärjestys on vaihdettavissa ainoastaan lisäasennuksille ja kolmansien osapuolten asennuksille.

Kohdekonfiguraation elementtien ominaisuuksia voidaan muuttaa valitun elementin ominaisuusnäkymästä kuvion 47 mukaisesti.



KUVIO 47. Konfiguraatioelementin ominaisuusnäkymä

Valmis konfiguraatiokuvaus rakennetaan installeripaketiksi ”Build”-komennolla, jolloin ohjelma suorittaa sivulla 61 esitetyn kuvion 42 mukaisen sekvenssin.

Tapauksissa, joissa valmiista konfiguraatiota halutaan laajentaa asennuspaketeilla muista konfiguraatioista, uuden konfiguraation luomisen sijasta laajennettava konfiguraatio voidaan avata kohdekonfiguraatioksi, tehdä siihen tarvittavat lisäykset ja rakentaa se Installeripaketiksi.

5.6.4 Konfiguraation tarkistaminen

Konfiguraatio tarkistetaan tarkistustyökalulla aina rakentamisen jälkeen automaattisesti ja löytyneet virheet sekä varoitukset raportoidaan ohjelman tulosikkunaan. Automaattinen tarkistus myös korjaa ristiriitaiset versionumerot ja tuotekoodit msi-asennuspakettien vastaavilla. Automaattisen tarkistuksen lisäksi sekä lähde- että kohdekonfiguraatiot voidaan tarkistaa käsin ”Validate”-komennolla, joka tarkistaa konfiguraation eheyden ja raportoi löytyneistä epäkohdista, mutta ei korjaa löytyneitä virheitä.

5.6.5 Konfiguraation lataaminen ja tallentaminen

Koska kohdekonfiguraatioksi voidaan avata myös valmiita konfiguraatioita, tallennus toteutettiin varmuuden vuoksi siten, että tallennettavan tiedoston nimi on asetettava ennen tallentamista. Tällöin valmista konfiguraatiota ei voi vahingossa ylikirjoittaa. Tallentaminen luo konfiguraatiotiedoston, joka sisältää varsinaisten konfiguraatioarvojen lisäksi ohjelman käyttämät metatiedot, joita ei kirjoiteta rakennettavaan konfiguraatiotiedostoon. Metatiedot erotetaan varsinaisista konfiguraatioarvoista kuvion 48 mukaisesti etuliitteellä //@.

```
[PKG1]
PackageName=SampleMainAppV1000.msi
PackageType=MSI
PackageDirectory=Main
Version=1.0.0.0
BackwardCompatibleToVersion=1.0.0.0
UpgradeCode={FC960CF0-3B10-4EF0-A905-A31F598B72D0}
ProductGUID={7C6C214D-04D5-4511-B4BD-62D744917DA7}
UninstallMode=ALWAYS
RequiredDiskSpace=0
//@sourcePath=D:\WORK\Installer2\TestCases\V1\Packages\Main
//@installationType=1
//@IncludeInBuild=True
```

KUVIO 48. Metatiedot konfiguraatiotiedostossa

5.6.6 Konfiguraatiohakemistojen avaaminen

Hakemistojen tarpeetonta selaamista vähentämään ohjelman käyttöliittymästä voi avata sekä lähdekonfiguraatioiden, että rakennettujen konfiguraatioiden hakemistot uuteen Windows Explorer -ikkunaan.

6 YHTEENVETO

Työssä oli tavoitteena toteuttaa helppokäyttöinen työkalusovellus Nokia Installerin asennuspakettien sisällön muokkaamiseen esimerkiksi matkapuhelinten mukana toimitettavan ohjelmistosisällön räätälöimiseksi. Asennuksen toiminnan ja Nokia Installerin tarkoituksen selventämiseksi tutkimusosuudessa perehdyttiin ohjelmistoasennusten merkitykseen osana ohjelmistotuotantoa, ohjelmistojen asentamiseen yleisimmissä käyttöjärjestelmissä, sekä Nokia Installer -sovellukseen. Pääpaino oli selkeästi Windows-käyttöjärjestelmien ohjelmistoasennuksissa, koska Nokia Installer on käytettävissä ainoastaan Windowsissa.

Työosa toteutettiin alkuvuodesta 2008, jonka jälkeen työkalu oli tarkoitus ottaa käyttöön. Suunnitelmat kuitenkin muuttuivat ja työkalusovellus jäi käytännössä hyväksi harjoitukseksi. Projektin toiminnalliset tavoitteet kyllä täyttyivät ja sovellus saatiin valmiiksi projektiaikataulussa, joskin päätoiminnallisuus saatiin toteutettua vasta projektin lopulla. Koodirivejä kertyi kaikkiaan n. 12000, joten ohjelmointiosa oli odotettua isotöisempi. Toteutusteknisistä ongelmista johtuvia seisahduksia tuli kuitenkin vain muutama. Enimmäkseen komentorivipohjaisia työkalusovelluksia C++:lla ohjelmoineena C#:n tehokkuus ja helppous vakuuttivat ja työstä sai arvokasta kokemusta .NET -ohjelmoinnista. Kaiken kaikkiaan työstä jäi onnistumisen tuntu, vaikka sovellusta ei otettukaan käyttöön.

Yksi sovelluksen mahdollisista jatkokehitysideoista on sen kehittäminen täysiveriseksi konfiguraatioeditoriksi, jossa konfiguraatioita on mahdollista luoda tyhjästä lisäten siihen uusia ominaisuuksia ja asennuksia. Sovelluksen jatkokehittäminen on kuitenkin riippuvainen omista resursseistani ja motivaatiosta, ei projektiluonteisesta työstä, mitä alkuperäinen toteutus oli.

LÄHTEET

Baker, B. 2001. The official InstallShield for Windows Installer Developer's guide. New York: M&T Books.

Baker, B. 2002. Getting started with InstallShield Developer and Windows Installer Setups. Schaumburg: InstallShield Press.

Baker, B. 2004. Practical Windows Installer Solutions. Schaumburg: InstallShield Press.

Apple. 2005. Bundle Programming Guide [verkkojulkaisu]. [viitattu 16.9.2008]. Saatavissa: http://developer.apple.com/documentation/CoreFoundation/Conceptual/CFBundles/CFBundles.pdf

Apple. 2006. Software Delivery Guide [verkkojulkaisu]. [viitattu 16.9.2008]. Saatavissa: http://developer.apple.com/documentation/DeveloperTools/Conceptual/SoftwareDistribution/SoftwareDeliveryGuide.pdf

Apple. 2007. PackageMaker User Guide [verkkojulkaisu]. [viitattu 16.9.2008]. Saatavissa: http://developer.apple.com/documentation/DeveloperTools/Conceptual/PackageMakerUserGuide/PackageMaker_UserGuide.pdf

Microsoft. 2006. Certified for Windows Vista Requirements. [verkkojulkaisu]. Microsoft [viitattu 18.1.2009.] Saatavissa: http://download.microsoft.com/download/8/e/4/8e4c929d-679a-4238-8c21-2dcc8ed1f35c/Windows%20Vista%20Software%20Logo%20Spec%201.1.doc

Microsoft. 1999. Windows Installer Service Overview [verkkojulkaisu]. Microsoft [viitattu 3.10.2008]. Saatavissa: http://download.microsoft.com/download/f/7/7/f777da84-f82d-4b90-a597-e329e09032b0/WIS-Pro.doc

Microsoft. 2008. Windows Installer File Name Extensions [verkkojulkaisu]. Microsoft [viitattu 17.11.2008]. Saatavissa: http://msdn.microsoft.com/en-us/library/aa372842(VS.85).aspx

Microsoft. 2008. Summary Information Stream [verkkojulkaisu]. Microsoft [viitattu 17.11.2008]. Saatavissa: http://msdn.microsoft.com/en-us/library/aa372044(VS.85).aspx

Microsoft. 2008. Installer Database [verkkojulkaisu]. Microsoft [viitattu 17.11.2008]. Saatavissa: http://msdn.microsoft.com/en-us/library/aa369395(VS.85).aspx

Microsoft. 2008. Standard Actions [verkkojulkaisu]. Microsoft [viitattu 17.11.2008]. Saatavissa: http://msdn.microsoft.com/en-us/library/aa372022(VS.85).aspx

Microsoft. 2008. Custom Actions [verkkojulkaisu]. Microsoft [viitattu 17.11.2008]. Saatavissa: http://msdn.microsoft.com/en-us/library/aa368066(VS.85).aspx

Microsoft. 2008. User Interface Levels [verkkojulkaisu]. Microsoft [viitattu 17.12.2008]. Saatavissa: http://msdn.microsoft.com/en-us/library/aa372391(VS.85).aspx

Debian. 2006. Debian Administration. Rolling your own Debian packages. [verkkojulkaisu]. [viitattu 13.1.2009.] Saatavissa: http://www.debian-administration.org/articles/336.

Acresso. 2009. InstallAnywhere Evaluation Guide. [verkkojulkaisu]. [viitattu 23.1.2009.] Saatavissa: http://kb.acresso.com/doc/DocumentRepository/Installation/InstallAnywhere/InstallAnywhere 2009/01_Public/Product_Manual/eg_ia_2009.pdf

Wikipedia. 2008. InstallAnywhere Wikipedia. [verkkojulkaisu] 6.12.2008. [viitattu 23.1.2009.] Saatavissa: http://en.wikipedia.org/wiki/InstallAnywhere.

Nokia. 2008. Nokia lyhyesti [verkkojulkaisu]. Nokia[viitattu 12.3.2009]. Saatavissa: http://www.nokia.fi/NOKIA_FINLAND_50/Nokia/About_Nokia/pdf/Nokia_lyhyesti_2007.pdf

Net Applications. elokuu 2008. Operating System Market Share [verkkojulkaisu]. [viitattu 10.9.2008]. Saatavissa: http://marketshare.hitslink.com/report.aspx?qprid=8

Gábor Deák Jahn. 2004-2008. Wix tutorial [verkkojulkaisu]. Tramontána [viitattu 4.10.2008]. Saatavissa: http://www.tramontana.co.hu/wix/

Nullsoft. 2008. NSIS Users Manual. [verkkojulkaisu]. [viitattu 18.1.2009.] Saatavissa: http://nsis.sourceforge.net/Docs/.

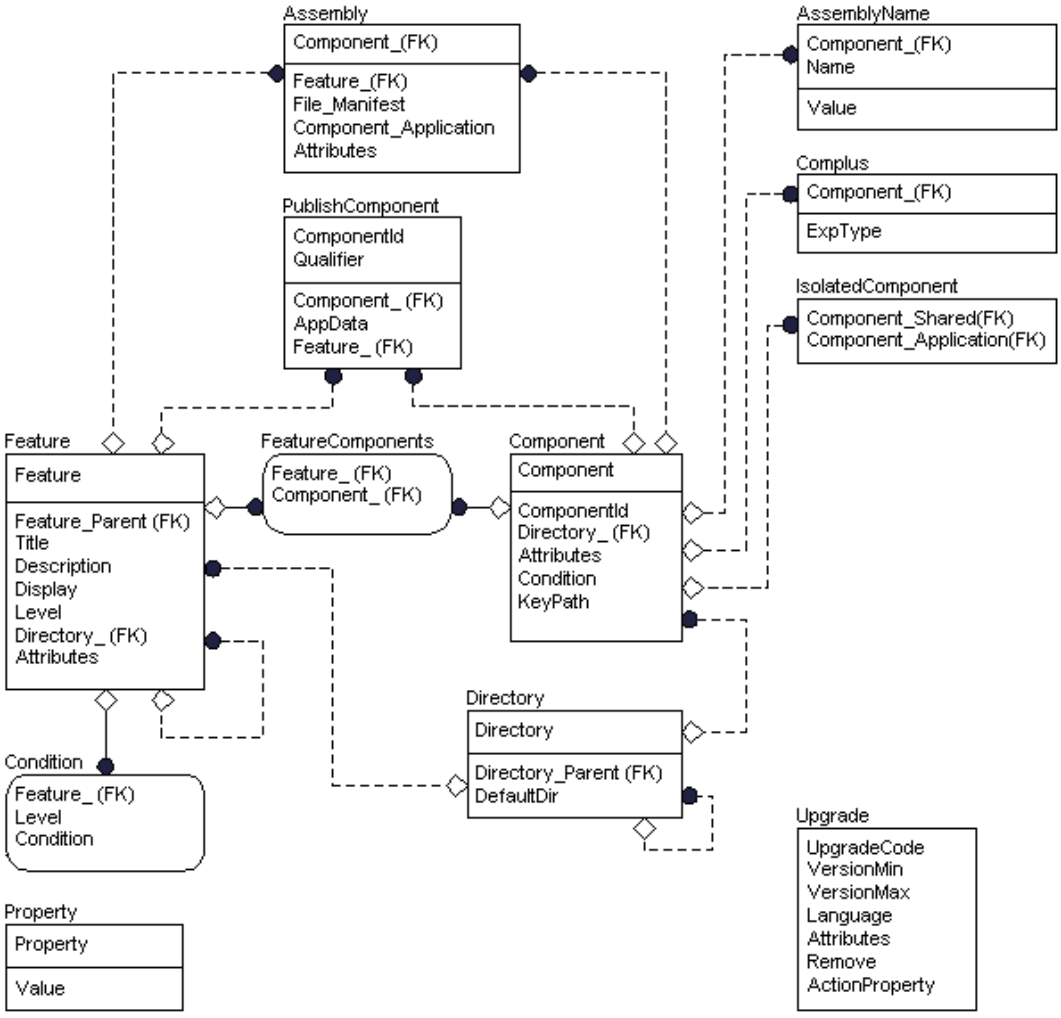
Wikipedia. 2008. Ubuntu - Wikipedia. [verkkojulkaisu]. [viitattu 11.1.2009.] Saatavissa: http://fi.wikipedia.org/wiki/Ubuntu.

Ubuntu. 2008. Ubuntu Documentation. Adding, Removing and Updating Applications. [verkkojulkaisu]. [viitattu 11.1.2009.] Saatavissa: https://help.ubuntu.com/8.10/add-applications/C/index.html

Symantec. 2006. WiseScript editor reference. [verkkojulkaisu]. [viitattu 24.1.2009.] Saatavissa: http://www.altiris.com/upload/wisescripteditor_005.pdf.

LIITTEET

Ydintaulut



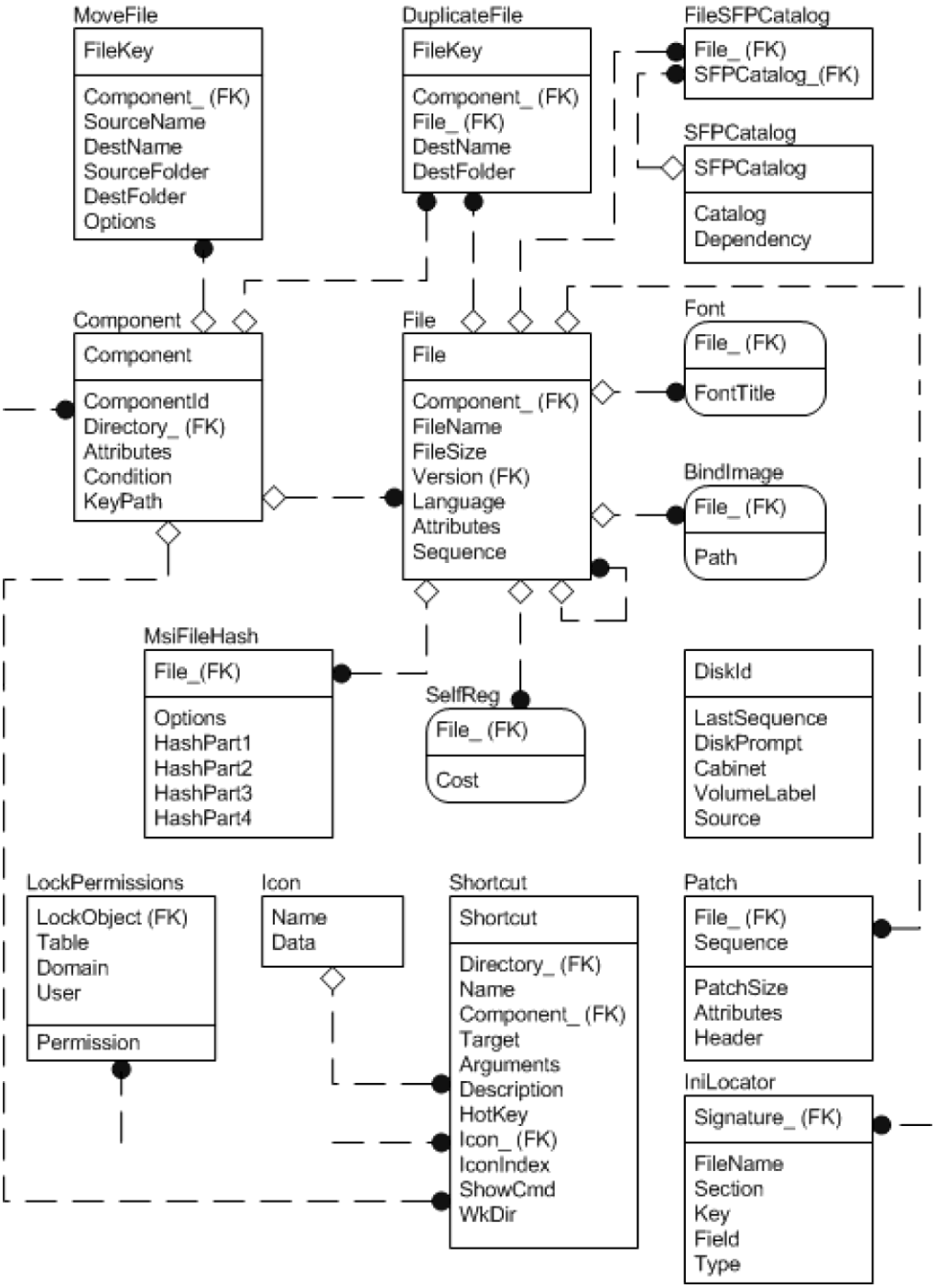
KUVIO 49. Ydintaulujen ryhmä (Installer database 2008)

Ydintaulut

TAULUKKO 5. Ydintaulujen ryhmä (Installer database 2008)

Taulu	Kuvaus
Feature	Sisältää asennukseen kuuluvat ominaisuudet.
Condition	Sisältää ominaisuuksien ehdolliset lauseet, jotka määrittävät, asennetaanko tietty ominaisuus vai ei.
FeatureComponents	Sisältää ominaisuuksiin kuuluvat komponentit.
Component	Sisältää asennukseen kuuluvat komponentit.
Directory	Sisältää asennuksen aikana tarvittavat hakemistot.
PublishComponent	Sisältää ominaisuudet ja komponentit, joita voidaan käyttää muista sovelluksista.
MsiAssembly, MsiAssemblyName	Määrittelee asennettavien .NET- ja Win32 -assemblyjen asetukset.
Complus	Sisältää COM+ -sovellusten asentamiseen tarvittavia tietoja.
IsolatedComponents	Assosioi ”Component_Application” (yleensä .exe -tiedosto) -sarakkeessa määritellyn komponentin ”Component_Shared” (yleensä .dll -tiedosto) -sarakkeessa määriteltyyn komponenttiin.
Upgrade	Sisältää ohjelmiston päivitykseen tarvittavia tietoja.

Tiedostotaulut



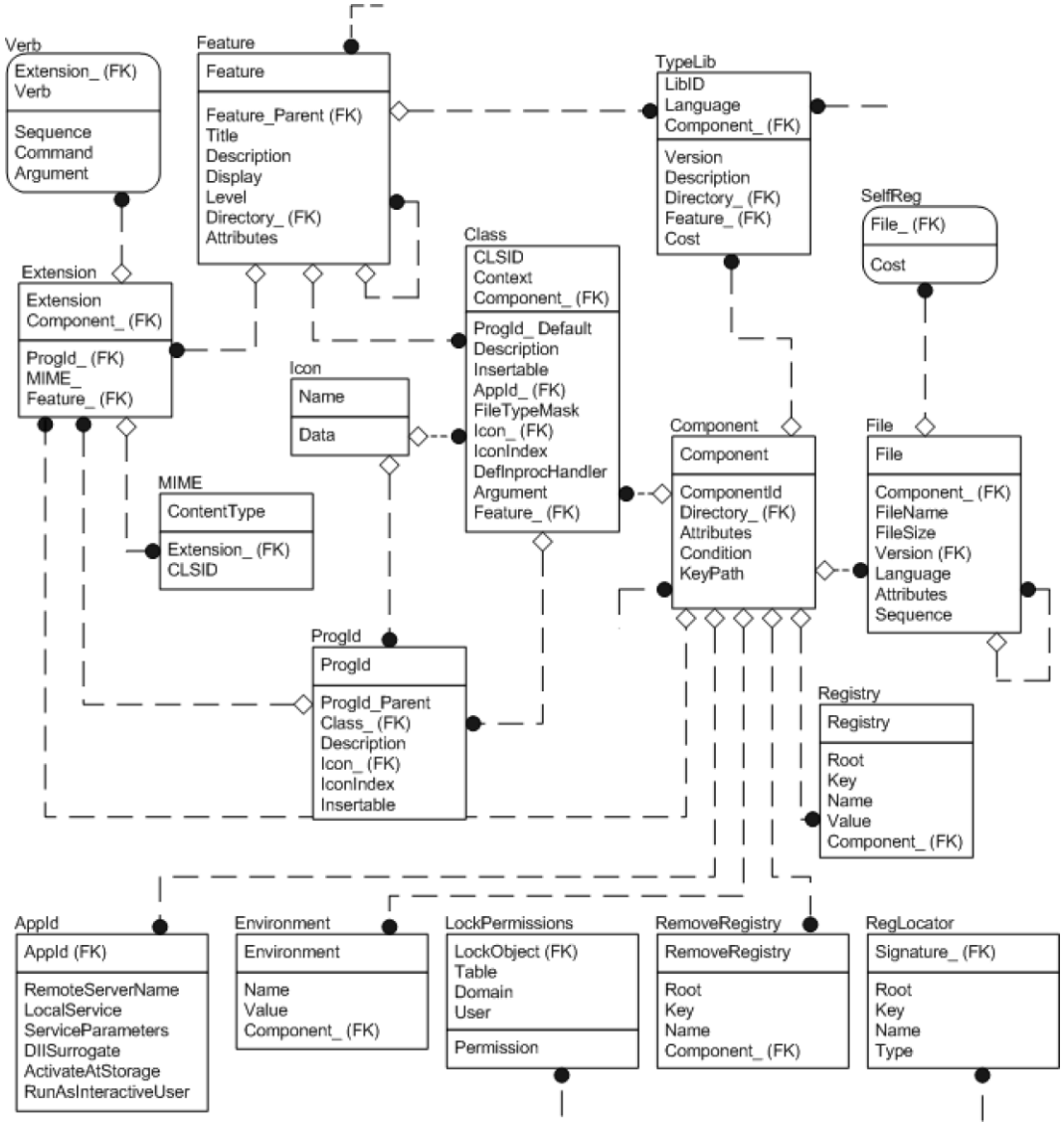
KUVIO 50. Tiedostotaulujen ryhmä (Installer database 2008)

Tiedostotaulut

TAULUKKO 6. Tiedostotaulujen ryhmä (Installer database 2008)

Taulu	Kuvaus
File	Sisältää asennukseen kuuluvat tiedostot. Tiedosto kuuluu aina jonkin komponentin alaisuuteen, joten taulusta on viite Component-tauluun.
RemoveFile	Sisältää tiedostot, jotka RemoveFiles -toiminto poistaa.
Font	Sisältää fonttitiedostot, jotka asennus rekisteröi järjestelmään.
SelfReg	Sisältää listan rekisteröitävistä COM-tiedostoista.
Media	Sisältää asennuksen lähdetiedostojen (CAB) sijainnin.
MoveFile	Sisältää asennuksen aikana siirrettävät tiedostot.
DuplicateFile	Sisältää asennuksen aikana monistettavat tiedostot. Tiedosto voidaan monistamalla sijoittaa useaan eri hakemistoon kohdejärjestelmässä.
IniFile	Sisältää .ini -tiedostot, joita muokataan asennuksen aikana, sekä .ini -tiedostoon kirjoitettavat arvot.
RemoveIniFile	Sisältää tiedot, jotka asennuksen tulee poistaa .ini -tiedostosta.
Environment	Sisältää ympäristömuuttujiin sijoitettavat arvot.
Icon	Sisältää tiedot asennuksen tarvitsemista kuvaketiedostoista.
FileSFPCatalog, SFPCatalog	Suojattujen tiedostojen ryhmään lisättävät tiedostot. Ei tuettu Windows 2000:ssa ja sen seuraajissa.
MsiFileHash	Sisältää lähdetiedostoista lasketut, 128-bittiset hajautusarvot.

Rekisteritaulut



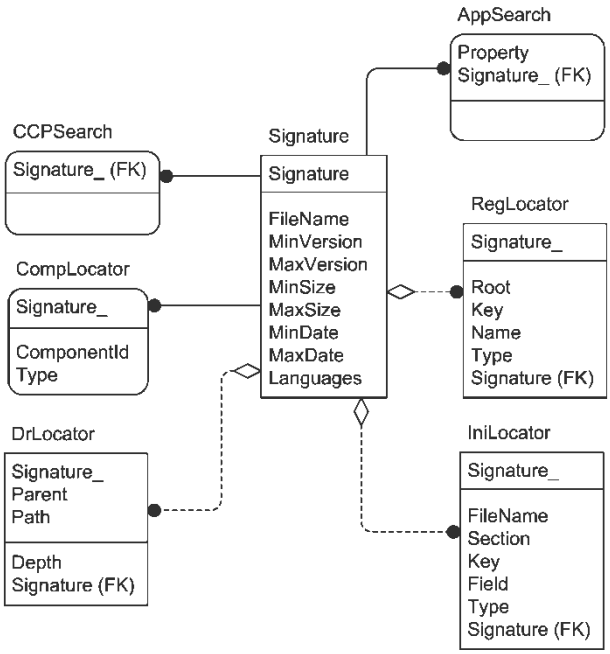
KUVIO 51. Rekisteritaulujen ryhmä (Installer database 2008)

Rekisteritaulut

TAULUKKO 7. Rekisteritaulut (Installer database 2008)

Taulu	Kuvaus
Extension	Sisältää asennettavan ohjelmiston käyttämät tiedostotunnisteet, jotka asennus rekisteröi järjestelmään.
Verb	Assosioi Extension -taulussa määriteltyihin tiedostotunnisteisiin komentorivejä vastaavat kuvaukset. Esimerkiksi tunniste .txt voidaan assosioida asennettavaan ohjelmaan X siten, että Windows Explorer näyttää .txt -tiedoston ponnahdusvalikossa vaihtoehdon ”avaa ohjelmassa X”. Valinta käynnistää ohjelman X komennnolla X ”%1”, jossa %1 korvautuu valitun .txt -tiedoston sijainnilla.
TypeLib	Sisältää tiedot jotka kirjoitetaan rekisteriin tyyppikirjastoja rekisteröitäessä.
MIME	Assosoi MIME-tyypin CLSID:hen tai tiedostotunnisteeseen.
SelfReg	Sisältää tiedostot, jotka Windows Installer rekisteröi kutsumalla tiedoston omaa DllRegisterServer -funktiota. Toiminto on tuettu taaksepäin yhteensopivuussyistä, eikä sen käyttäminen ole suositeltavaa uusissa asennuksissa.
Class	Käytetään COM -objektien luokkatunnisteiden (CLSID) rekisteröintiin.
ProgId	Assosioi ohjelmatunnisteet (ProgId) luokkatunnisteisiin (CLSID)
AppId	Käytetään DCOM-objektien asetusten rekisteröintiin.
Environment	Sisältää ympäristömuuttujiin sijoitettavat arvot, jotka Windows 2000:sta lähtien kirjoitetaan myös rekisteriin.
Registry	Sisältää kaiken muun rekisteriin kirjoitettavan informaation, mitä ei voida kirjoittaa muiden rekisteritaulujen kautta.
RemoveRegistry	Sisältää rekisteristä asennuksen aikana poistettavat tiedot.

Etsintätaulut

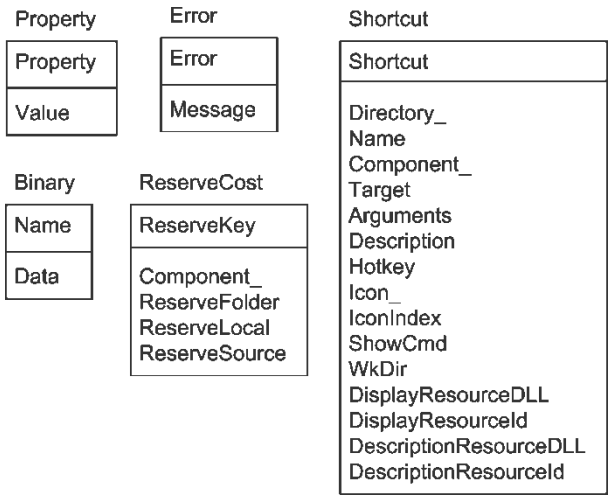


KUVIO 52. Etsintätaulujen ryhmä (Installer database 2008)

TAULUKKO 8. Etsintätaulut (Installer database 2008)

Taulu	Kuvaus
Signature	Sisältää tiedoston yksilöivän tunnuksen.
AppSearch	Sisältää muuttujan, joka asetetaan etsittävän Signature-tunnuksen löytyessä.
CCPSearch	Sisältää listan Signature-tunnuksia, joita käytetään tarkistettaessa ohjelmiston yhteensopivuutta päivitystapauksessa.
CompLocator	Sisältää tietoja tiedoston tai hakemiston etsimiseen Windows Installerin aiemmin asentamista komponenteista.
DrLocator	Sisältää tietoja tiedoston tai hakemiston etsimiseen kiintolevyn hakemistoista.
IniLocator	Sisältää tietoja .ini -tiedoston etsimiseksi. Tiedoston on sijaittava Windows-hakemistossa.
RegLocator	Sisältää tietoja tiedoston tai hakemiston etsimiseen rekisteristä.

Informaatiotaulut

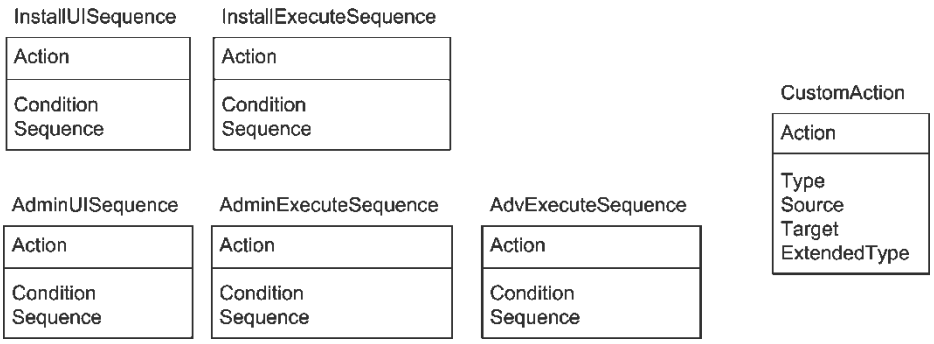


KUVIO 53. Informaatiotaulut (Installer database 2008)

TAULUKKO 9. Informaatiotaulut (Installer database 2008)

Taulu	Kuvaus
Property	Sisältää kaikki asennuksen muuttujat.
Binary	Sisältää binääritiedostoja, kuten kuvia, animaatioita, ikoneita. Mukautettuja toimintoja toteuttavat tiedostot talletetaan binääritauluun.
Error	Sisältää virhekoodeihin perustuvien virheilmoitusten mallipohjat.
Shortcut	Sisältää tietoja pikakuvakkeiden luontia varten.
ReserveCost	Valinnainen taulu, jolla voidaan varata asentamiseen tarvittavaa kiintolevytilaa.

Asennusproseduuritaulut

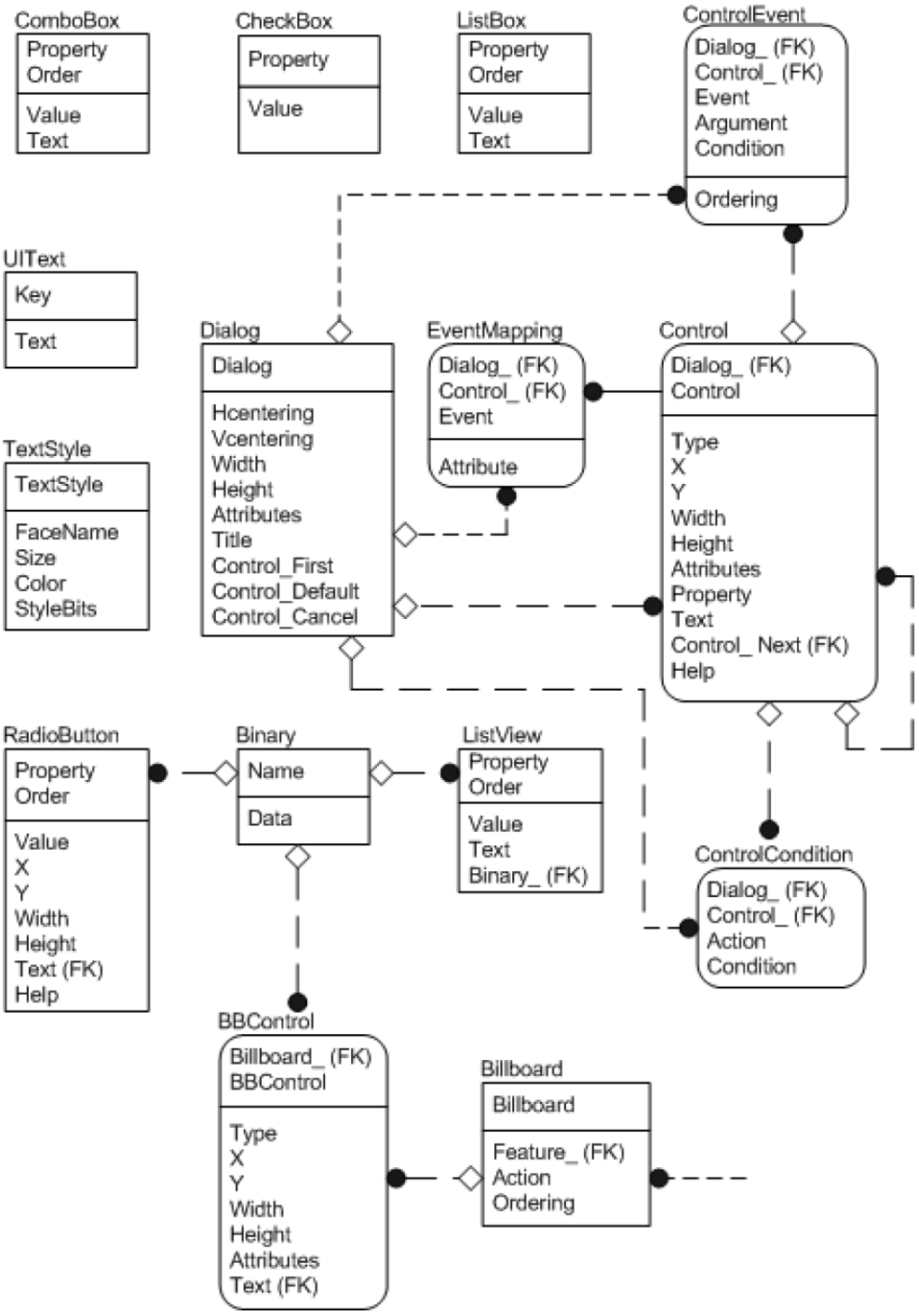


KUVIO 54. Asennusproseduuritaulut (Installer database 2008)

TAULUKKO 10. Asennusproseduuritaulut (Installer database 2008)

Taulu	**Kuvaus**
InstallUISequence	Sisältää toimintoja, jotka suoritetaan päätoiminnon INSTALL aikana, mikäli Windows Installerin sisäinen käyttöliittymä on kokonaan tai rajoitetusti käytössä (käyttöliittymän tila= full/reduced). Muutoin taulun määrittämät toiminnot ohitetaan.
InstallExecuteSequence	Sisältää toimintoja, jotka suoritetaan päätoiminnon INSTALL aikana.
AdminUISequence	Sisältää toimintoja, jotka suoritetaan päätoiminnon ADMIN aikana, mikäli Windows Installerin sisäinen käyttöliittymä on kokonaan tai rajoitetusti käytössä. Muutoin taulun määrittämät toiminnot ohitetaan.
AdminExecuteSequence	Sisältää toimintoja, jotka suoritetaan päätoiminnon ADMIN aikana.
AdvtExecuteSequence	Sisältää toimintoja, jotka suoritetaan päätoiminnon ADVERTISE aikana. Sallitut toiminnot on ennalta määritetty eikä mukautettuja toimintoja voi käyttää.
CustomAction	Sisältää mukautettujen toimintojen määrittelyt.

Käyttöliittymätaulut



KUVIO 55. Käyttöliittymätaulut (Installer database 2008)

TAULUKKO 11. Mukautetut toiminnot. (Custom Actions 2008)

<table>
<tr><th>Tyyppi</th><th>Kuvaus</th></tr>
<tr><td>1</td><td>Binääritauluun talletettu, C/C++ -kielellä kirjoitettu dll-kirjasto, josta vietyä funktiota kutsutaan. Funktioesittely on muotoa:

<pre>extern "C"{
  UINT __stdcall FunktioNimi(MSIHANDLE hInstall)
}</pre>
Funktion paluuarvona tulee olla joku seuraavista:

<table>
<tr><td>ERROR_FUNCTION_NOT_CALLED</td><td>Toimintoa ei suoritettu.</td></tr>
<tr><td>ERROR_SUCCESS</td><td>toiminto onnistui.</td></tr>
<tr><td>ERROR_INSTALL_USEREXIT</td><td>Käyttäjä peruutti suorituksen.</td></tr>
<tr><td>ERROR_INSTALL_FAILURE</td><td>Kriittinen virhe.</td></tr>
<tr><td>ERROR_NO_MORE_ITEMS</td><td>Jäljellä olevat kohteet ohitetaan, ei virhe.</td></tr>
</table>
Funktio määritellään .def -tiedostossa tai linkitetään /EXPORT -määreellä.</td></tr>
<tr><td>2</td><td>Suoritettava exe-tiedosto binääritaulussa. Tiedostolle voidaan välittää komentoriviparametreja. Mikäli paluuarvon käsittely on sallittu, ainoa hyväksyttävä paluuarvo on 0, muut arvot tulkitaan asennuksen keskeyttäväksi virheeksi.</td></tr>
<tr><td>5</td><td>JScript-tiedosto binääritaulussa. Tiedostoa voidaan kutsua suoraan tai jotain siinä määritettyä funktiota. Muun kuin onnistuneen paluuarvon palauttamiseksi on käytettävä funktiokutsua. Mahdolliset palautusarvot ovat tällöin:

<table>
<tr><td>0</td><td>Toimintoa ei suoritettu.</td></tr>
<tr><td>IDOK</td><td>toiminto onnistui.</td></tr>
<tr><td>IDCANCEL</td><td>Käyttäjä peruutti suorituksen.</td></tr>
<tr><td>IDABORT</td><td>Kriittinen virhe.</td></tr>
<tr><td>IDRETRY</td><td>Toiminto keskeytettiin jatkettavaksi myöhemmin.</td></tr>
<tr><td>IDIGNORE</td><td>Jäljellä olevat kohteet ohitetaan.</td></tr>
</table></td></tr>
</table>

TAULUKKO 12. Mukautetut toiminnot (Custom Actions 2008).

Tyyppi	Kuvaus
6	VBScript-tiedosto binääritaulussa. Käytetään kuten tyyppiä 5.
17	Asennettava dll-tiedosto. Toiminta kuten tyypillä 1, mutta toiminto on käytettävissä vasta tiedoston asentamisen jälkeen.
18	Asennettava exe-tiedosto. Toiminta kuten tyypillä 2, mutta toiminto on käytettävissä vasta tiedoston asentamisen jälkeen.
19	Asennuksen keskeyttävän virheilmoituksen näyttävä toiminto. Näyttää Error-tauluun talletetun virheilmoituksen, jonka jälkeen asennus keskeytetään.
21	Asennettava JScript-tiedosto. Toiminta kuten tyypillä 5, mutta toiminto on suoritettavissa vasta tiedoston asentamisen jälkeen.
22	Asennettava VBScript-tiedosto. Toiminta kuten tyypillä 6, mutta toiminto on suoritettavissa vasta tiedoston asentamisen jälkeen.
34	Hakemistoviittaus exe-tiedostoon. Directory-taulu sisältää hakemiston, jossa tiedosto sijaitsee. Toiminta on muuten kuten tyypillä 2.
35	Hakemistonmuuttujan asettava toiminto.
37	Sekvenssitauluun talletettu JScript-lähdekoodi.
38	Sekvenssitauluun talletettu VBScript-lähdekoodi.
50	Property-tauluun talletettu exe-tiedoston polku. Toiminta on muuten kuten tyypillä 2.
51	Muuttujan asettaminen
53	Muuttujaan talletettu JScript-lähdekoodi
54	Muuttujaan talletettu VBScript-lähdekoodi