Kretschmer, Peter

Entwicklung eines automatischen Updatesystems für Softwareanwendungen

eingereicht als

Diplomarbeit

an der

Hochschule Mittweida (FH)

University of Applied Sciences

Fachbereich Elektro- und Informationstechnik

Dresden, 2012

Erstprüfer : Prof. Dr.-Ing. Wilfried Schmalwasser

Zweitprüfer: Michael Unglaub

vorgelegte Arbeit wurde verteidigt am:

Bibliographische Beschreibung:

Kretschmer, Peter:

Entwicklung eines automatischen Updatesystems für Softwareanwendungen. -2012. -81S.

Dresden, Hochschule Mittweida (FH), Fachbereich Elektro- und Informationstechnik, Diplomarbeit, 2012

Referat:

Ziel der Diplomarbeit ist die Entwicklung eines Updatetools, um einen breite Palette an Softwareprodukten eines Herstellers zu aktualisieren. Dazu findet eine Analyse von Installationsverfahren, Softwareverteilung und Softwaremanagement statt. Es werden verschiedene Arten von Updates verglichen und technische Probleme behandelt. Auch wird auf die Sicherheit bei Updates eingegangen. Zum Schluss werden die Entwicklung und die Funktionalität des Updatetools erläutert.

Inhaltsverzeichnis

Fachbegriffe und Abkürzungen

Array: Datenfeld

Cloud: onlinebasierte Speicher- und Serverdienste

Compiler: Programm, das Code aus Programmiersprachen in Maschinensprache übersetzt

Content: Inhalte

Deployment: Softwareverteilung

DLL : Dynamic Link Library, Dynamische Verbindungsbibliothek

Domain: zusammenhängender Teilbereich des hierarchischen Domain Name System (DNS)

GUI : Graphical User Interface, Grafische Benutzeroberfläche

Header: Kopf, Kopfdaten

Host: Computer, der Dienste in einem Rechnernetz zur Verfügung stellt

Idle: untätig

Kompilierung: Übersetzung von Programmcode in Maschinensprache

Mode: Modus

Multithreading: Gleichzeitiges Arbeiten mehrerer Threads in einem Prozess

Plugin: Erweiterungsmodul

Tool: kleine Anwendungssoftware oder Dienstprogramm

Tweak: Tool zur Optimierung, Leistungssteigerung

Unit: hier: Delphi-kompatibles Bibliotheksmodul

WAN : Wide Area Network, großräumiges Netz

0. Einleitung

Unter einem Softwareupdate versteht man die Wartung und Pflege einer Softwareanwendung. Dazu gehört die Behebung von Fehlern, die Stabilisierung des Laufverhaltens der Software sowie Steigerungen der Performance und das Einspielen von neuen Funktionen. Die Aktualisierung sollte weitgehend automatisiert sein, sodass nur ein geringer Mehraufwand für den Nutzer entsteht.

Ziel dieser Arbeit ist die Entwicklung eines Updatetools für Softwareanwendungen und die Analyse von vorhandenen Updatesystemen. Zuerst sollen Grundlagen der Paketierung und Installationsmechaniken aufgezeigt werden. Danach sollen Methoden und Verfahren zur Softwareverteilung behandelt werden und sicherheitsrelevante Fragen bearbeitet werden.

Zur Entwicklung des Updatetools sollen Maßnahmen zur Realisierung der Anforderungen erläutert und dazu Voraussetzungen und Problemstellungen diskutiert werden.

Zum Schluss soll das Updatetool in seiner Funktionalität erläutert werden, wobei die wichtigsten Aspekte der Kommunikation zwischen Clients und Server sowie die Funktionen des Tools und der serverseitigen Anwendung aufgezeigt werden sollen.

1 Updatesysteme

1.1 Differenzierung Updatesysteme

1.1.1 Das Internet

Das Softwareupdate stellt dem Nutzer einer Software Aktualisierungen bereit, die er sich auf verschiedenen Wegen besorgen kann. Heutzutage ist das Internet die geläufigste Plattform, um Updates runterzuladen und Programme auf dem Laufenden zu halten. Warum ist das so?

Das Internet bietet sich als Medium für Updates an, da es in seiner weltweiten Verbreitung alles kann, was selbst für größere Einspielungen notwendig ist. Im Jahr 2011 haben weltweit ca. 2,1 Milliarden Menschen Zugang zum Internet gehabt. Davon waren 591 Millionen im Besitz eines Breitbandanschlusses.[1]

Laut Daten der OECD aus den Jahren 2010 und 2011 existieren in etwa 86 Prozent der deutschen Haushalte ein PC, Notebook oder Tablet PC. Etwa 27 Millionen Haushalte verfügen über einen Breitbandinternetanschluss, das entspricht in etwa 75 Prozent. Die Tendenz ist steigend. Jedes Jahr erhöht sich die Anzahl der Anschlüsse um etwa 4 Prozent. 82,5 Prozent der Deutschen verfügen über einen Internetzugang. [2]

So ist es nicht verwunderlich, dass Softwareriesen wie *Microsoft*, *Adobe* oder *SAP* ihre Updates fast ausschließlich über das Internet veröffentlichen.

Selbst größere Updates können in immer kürzerer Zeit geladen werden. Die Download-Geschwindigkeit deutscher Anschlüsse liegt im Schnitt bei 5,92 MBit/s. Dabei befindet man sich allerdings weltweit nur im Mittelmaß. Spitzenreiter ist Japan. Auf den japanischen Inseln können die User im Schnitt mit sagenhaften 61 MBit/s Daten runterladen. Die USA befinden sich in dieser Statistik mit 4,8 MBit/s auch nur im Mittelmaß.[3]

Das Internet ist prädestiniert für die Herausgabe von Updates, da der größte Teil der Softwarenutzer über einen Anschluss mit genügender Bandbreite für die meisten Zwecke verfügt. Alternativ können Angebote zum Verschicken von CDs oder die Installation vor Ort vorkommen. Im Folgenden wird nur noch das Internet als hauptsächliches Transportmedium

[1] Vgl: http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/
[2] Vgl: http://www.oecd.org/dataoecd/19/46/34083096.xls;
http://www.oecd.org/dataoecd/22/15/39574806.xls;
http://www.oecd.org/dataoecd/22/11/39574765.xls;
http://www.oecd.org/dataoecd/19/45/34083073.xls
[3] Vgl: http://mareikewueste.files.wordpress.com/2010/01/cwinv.jpg

behandelt. Es ist günstig, schnell und komfortabel. Durch die Verbreitung werden andere Medien zunehmend unattraktiver. So auch für den Vertrieb und vor allem die Instandhaltung von Softwareprodukten.

1.1.2 Grundlagen

„Update [ˈʌpdɛit; das; englisch]: neue, überarbeitete Fassung oder Ausgabe eines EDV-Programms oder eines Artikels.“[4]

Im englischen lautet das Adjektiv zu Update *up to date*. Übersetzen kann man das mit *aktuell*, *auf dem Laufenden* oder *auf dem neuesten Stand*.[5]

Der Lebenszyklus von Software sieht im Wasserfallmodell nach der Implementierung und dem Test der Software den Betrieb und die Wartung auf einer Stufe (s. Abbildung 1).[6]

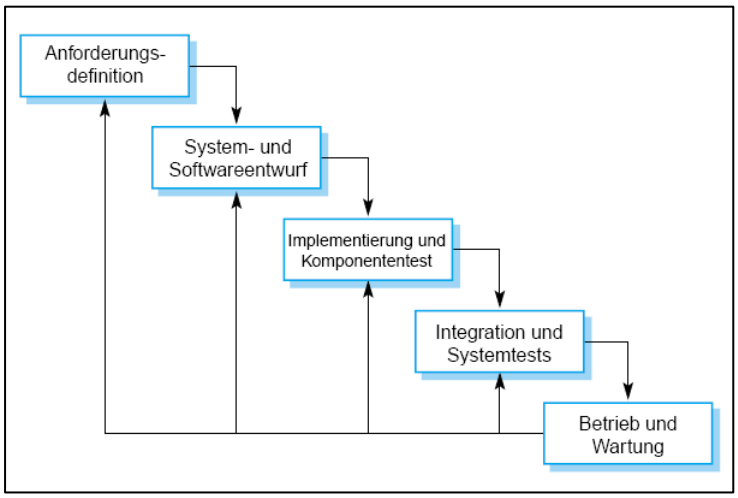

Abbildung 1: Der Softwarelebenszyklus

Jeder Softwareentwickler muss den Anspruch haben, seine Applikationen auf dem Laufendem zu halten. Andernfalls werden diese vom Kunden nicht auf lange Zeit akzeptiert und ersetzt. Er wird wahrscheinlich die Software des Herstellers meiden und im schlimmsten Fall seine Reputation diskreditieren. Um dem entgegen zu wirken, soll und muss der Entwickler regelmäßig Updates einspielen. Der Inhalt dieser Updates kann vielseitig sein und mehrere Zwecke erfüllen. Er kann technische Einflüsse auf das Programm haben oder ökonomische Ziele verfolgen.

[4] Quelle: http://www.wissen.de/lexikon/update?keyword=update
[5] Quelle: http://www.dict.cc/?s=up+to+date
[6] Quelle: http://www.tim-christmann.de/studium/images/wasserfallmodell.gif

1.1.3 Der Patch

„Patch [pætʃ] m. oder n.; - od. s, -s] : ⟨EDV⟩ (meist kostenlos zur Verfügung gestelltes) Softwareprogramm, das in einem vorhandenen Programm enthaltene Fehler od. Mängel beheben soll.“[7]

Der Patch wird in vorhandene Software eingespielt, um Fehler im Programmcode zu beheben. Er wird in diesem Zusammenhang auch *Bugfix* genannt. Oftmals werden Fehler erst während der Betriebsphase der Software erkannt und können mittels Patch behoben werden. Ist der Eingriff eilig, wird er als *Hotfix* bezeichnet. In diesem Fall wird der Patch nicht wie üblich während der Wartungsintervalle ausgeliefert, sondern steht sofort zur Verfügung. Dieses Verfahren wird bei groben Fehlern verwendet, die etwa beim letzten Update entstanden. Auch werden auf diese Weise schnell Sicherheitslücken geschlossen. So bietet zum Beispiel *Microsoft* regelmäßig Hotfixes zum Download[8] an, um Sicherheitslücken zu schließen oder um Angriffen vorzubeugen bzw. diese abzuwenden. In der Regel werden die Hotfixes dann noch weiter getestet und im nächsten *Windows Update* als Patch ausgeliefert.

1.1.4 Das Sicherheitsupdate

Sicherheitslücken entstehen durch Programmierfehler im Betriebssystem oder in Softwareanwendungen sowie durch ungenügenden Schutz durch Firewalls, Antivirensoftware oder anderen Sicherheitsmaßnahmen.

Diese Lücken werden durch *Exploits* ausgenutzt[9], mit denen man das System des Ziels manipulieren will. Werden solche Schwachstellen entdeckt, versucht man, sie in der Regel mit *Hotfixes* zu beheben.

1.1.5 Das Datenupdate

Hier wird kein Programmcode gepatcht sondern es werden inhaltliche Änderungen vorgenommen. So können beispielsweise Datenbanken überholt oder Konfigurationsdateien bzw. einfach nur Text ersetzt, gelöscht oder erweitert werden.

[7] Quelle: http://www.wissen.de/fremdwort/patch?keyword=patch
[8] Quelle: http://www.microsoft.com/en-us/download/default.aspx?
[9] Vgl.: Kapitel 1.9.1 Risiken und deren Enstehung

1.2 Paketierung und Installation

1.2.1 Grundlagen der Paketverwaltung

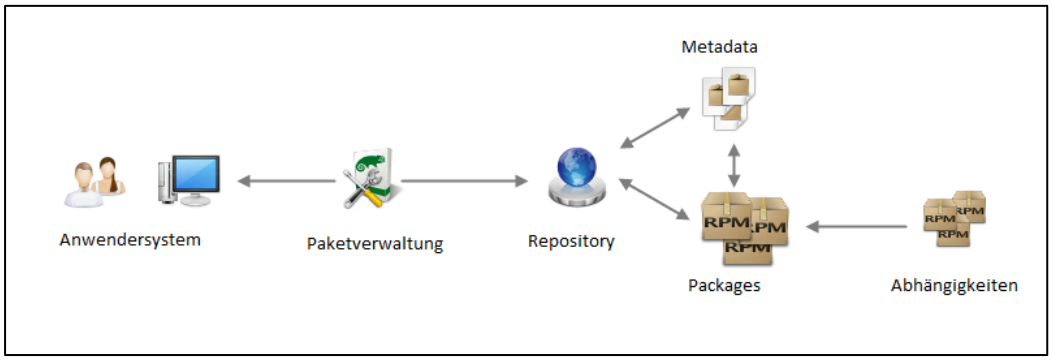

Abbildung 2: Im Repository werden alle Informationen für die Installation gesammelt

Ein Paketverwaltungssystem (englisch: *package management system*) stellt einem Betriebssystem Funktionen zur Installation, Deinstallation und zum Upgrade von Software bereit. Pakete oder Packages enthalten die zur Installation benötigten Dateien. Dazu können ausführbare Dateien, Skripte oder Mediadateien wie Audio-, Video- und Bilddateien gehören. Sie werden vom *Packager* erstellt, der die Regeln für die Installation festlegt und die Dateien archiviert. Zusätzlich kann der Packager noch Patches sowie die Anordnungen für deren Installation eintragen.

Ebenso trägt er auch die Abhängigkeiten (engl.: *dependencies*) in eine Liste ein. Das bedeutet, dass Pakete, die für die Installation des eigentlichen Pakets benötigt werden, angegeben werden. Diese Abhängigkeiten sind transitiv, d.h., wenn das Paket A zur Installation ein Paket B benötigt und das wiederum das Paket C, dann wird das Paket A auch das Paket C benötigen. Daher kann es vorkommen, dass manchmal viele Pakete benötigt werden, um nur ein einziges Programm zu installieren.

Bevor die Pakete nun installiert werden können, werden sie in einem Container zur Verfügung gestellt, dem *Repository.* Hier können zusätzlich zu den Programmpaketen Metadaten abgelegt werden. Diese können Dokumentationen zur Software oder Informationen über die Abhängigkeiten enthalten. Die Abhängigkeiten können dann automatisch von der Paketverwaltung heruntergeladen und installiert werden.

Die Paketverwaltung arbeitet typischerweise betriebssystemabhängig. Sie wird daher in den meisten Fällen vom Hersteller des Betriebssystems mitgeliefert und arbeitet speziell für dieses.

Es gibt aber auch einige Paketverwaltungen von Drittanbietern. Nachfolgend werden Paketverwaltungssysteme, auch Installer genannt, für die Betriebssysteme des Herstellers Microsoft behandelt. Um den Installertyp festzustellen, bringt neben der Lektüre des Handbuchs der Anwendung oft auch die Angabe der Parameter „/?“ oder „/help“ bei der Installation die gewünschte Antwort.

1.2.2 Unattended Setup und Skriptgesteuerte Installation

Man unterscheidet bei Installationsverfahren zwischen **Unattended Setup** und **Skriptgesteuerter Installation**. Unattended (deutsch: unbeaufsichtigt) bezeichnet eine weitgehend automatisierte Installation, bei der vorher festgelegte Automatismen die Installation steuern und nur geringe Eingriffsmöglichkeiten des Nutzers zulassen. Meist werden einige Parameter zur Steuerung der Installation (z.B. */quiet*) zugelassen, bei der eine *Silent Installation* ohne GUI ausgeführt wird. Ein Beispiel hierfür ist die Installation von Windows, die weitgehend automatisch abläuft.

Die skriptgesteuerte Installation ähnelt dem Unattended Setup, da sie, oberflächlich gesehen, ebenfalls weitgehend automatisch abläuft. Näher betrachtet weist sie sich allerdings durch einen deutlichen Unterschied aus, denn im Gegensatz zum Unattended Setup stellt das Skript nur eine Anreihung diverser Befehle dar, die nur Eingaben durch den Nutzer simulieren. Dadurch entsteht mehr Flexibilität beim Installationsvorgang. Jedoch ist das Scripting meist mit mehr Aufwand bei der Erstellung der Installation verbunden. Als Beispiel können die Installationen des Nullsoft Scriptable Install System[10] genannt werden.

1.2.3 Der Windows Installer

Es gibt unter Windows kein vereinheitlichtes Installationsverfahren. Der *Microsoft Software Installer* legt sicherlich den Standard fest, jedoch sind noch einige andere Installer unter Windows präsent und werden regelmäßig verwendet. Seit Windows 2000 wird er mit dem Betriebssystem mitgeliefert, kann aber für Windows 95 und 98 nachinstalliert werden. Heute wird er *Windows Installer genannt.*

[10] Vgl.: Kapitel 1.3.4 Das Nullsoft Scriptable Install System (NSIS)

Für die Installation, Deinstallation und Reparatur der Pakete läuft in der Laufzeitumgebung des Windows Installer ein Windows Systemdienst, der *msiserver*. Er führt die notwendigen Installationsdateien mit der Endung **.msi (Microsoft-Software-Installation)* aus. In diesen Dateien befinden sich Installationsbeschreibungen, die tabellarisch in relationalen Datenbanken angeordnet sind. Damit erfolgt die Installation regelbasiert und nicht prozedural[11].

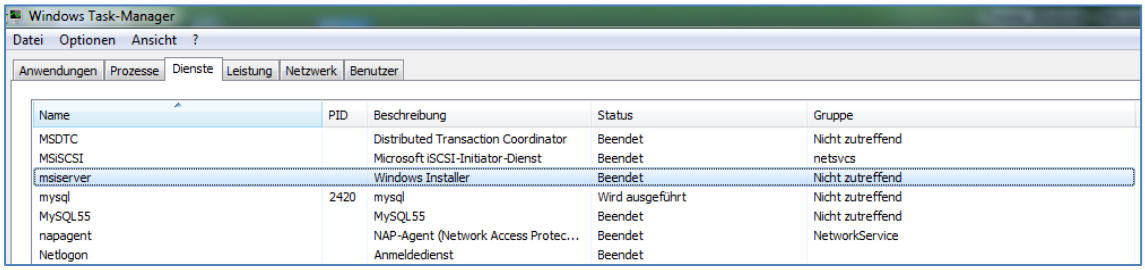

Abbildung 3: Der msiserver Dienst wird vom Windows-Systemdienst aufgerufen, wenn ein Prozess die WinAPI Funktion StartService aufruft.

1.2.3.1 Funktionen

Der Windows Installer verfügt über verschiedene Features. Wir beschränken uns im Folgenden auf die wichtigsten:

Mittels **Transformation** kann die Installationsroutine für eine Anwendung verändert werden. Hier werden Standardvorgaben angepasst und in speziellen Transforms-Dateien (**MST**-Dateien) gespeichert. Sie werden beim Start der Installation hinzugefügt, wobei ein Original-Installationspaket mehrere MST Dateien besitzen kann, ohne selbst verändert zu werden.

Mit dem Tool Orca aus dem Microsoft Windows SDK (Source Developement Kit) können schnell MSI Dateien bearbeitet werden. Ebenfalls kann es Transforms erstellen. Außerdem bieten Drittanbieter, wie *Flexera* (*InstallShield*) das Programm *Adminstudio*[12], Tools zum Erstellen der MST an. Diverse Softwarehersteller bieten Applikationen zur Erstellung von Transform Files speziell für ihre Produkte an, wie zum Beispiel den *Custom Installation Wizard* im *Microsoft Office Resource Kit*[13] von *Microsoft*.

[11] Prozedurale Programmierung ist der Ansatz Programme in kleinere Teilprobleme (oder genauer: Aufgaben) die als Prozeduren bezeichnet werden aufzuspalten. (Vgl. http://www.uni-protokolle.de/Lexikon/Prozedurale_Programmierung.html)
[12] http://www.flexerasoftware.com/products/adminstudio.htm
[13] Vgl.: http://technet.microsoft.com/de-de/library/cc303401.aspx

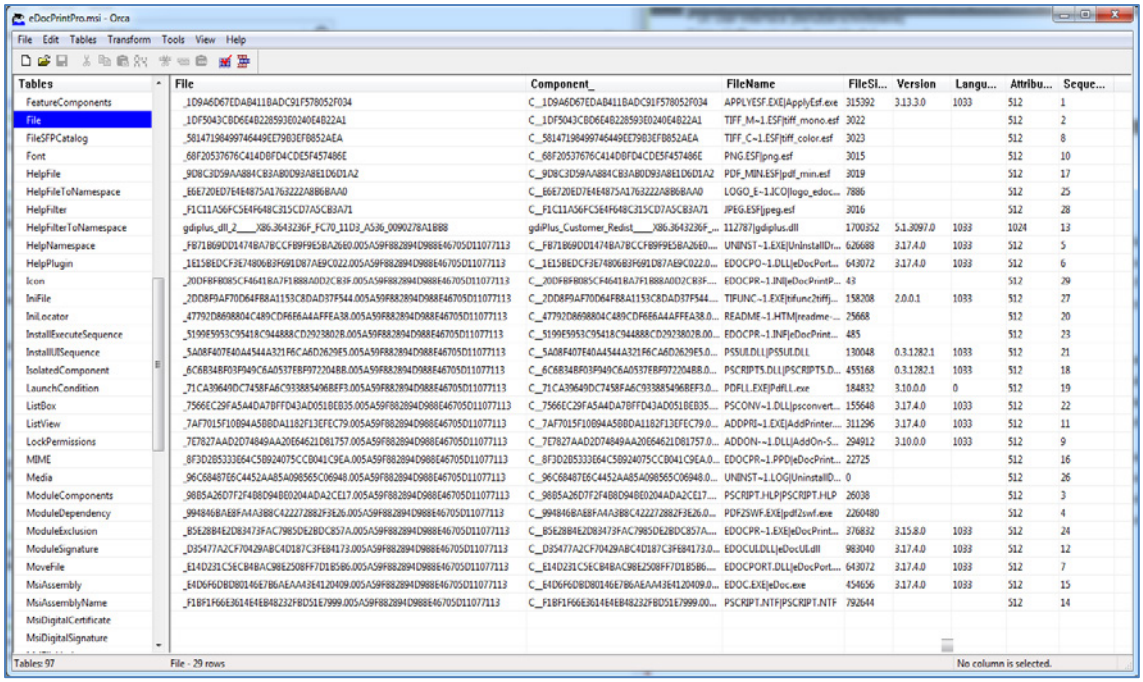

Abbildung 4: In der Tabelle File des Installer Pakets für den Treiber eDocPrintPro sind die Dateien enthalten, die installiert werden. Mit dem Orca Tool können die Tabelleneinträge angelegt, bearbeitet und gelöscht werden.

Um Platz auf der Festplatte zu sparen und Anwendungen etwas an Komplexität zu nehmen, kann eine **Installation bei Bedarf** stattfinden. Hier können beispielsweise Sprachpakete vorerst nicht installiert, aber später manuell hinzugefügt werden.

Die **Installation ohne UI**[14] kann über das Programm *msiexec* gestartet werden. Dabei können der Installation diverse Parameter übergeben werden. Der Windows Installer bietet Möglichkeiten, um die installierten Anwendungen per **Update** aktuell zu halten.

Ebenso können **Reparaturen** getätigt werden. Hier prüft der Installer während der Laufzeit der Applikation, ob Dateien beschädigt sind oder nicht existieren. In diesem Fall würden die entsprechenden Dateien von der Installationsquelle neu eingespielt werden.

Neben der **Deinstallation** von Anwendungen gibt es die Option des **Rollbacks**, welche bei fehlgeschlagener Installation den Systemzustand vor der Installation wiederherstellt, da sonst Dateien oder Ordner auf dem System zurückbleiben könnten. Wird zum Beispiel eine Installation während der Dateiübertragung abgebrochen, kann es zu Dateninkonsistenzen kommen, so dass Datenbanken falsche Werte zurückliefern oder Anwendungen Fehlermeldungen zurückgeben würden. Das Rollback kann diese Fehler eliminieren, indem es sämtliche Änderungen, die bei der Installation vorgenommen wurden, zurücknimmt.

[14] UI: User Interface (Benutzerschnittstelle)

Die **Ausführung vom Quellmedium** ist ein Feature, das Verweise auf mehrere Installationsquellen erkennt. So kann etwa bei einem Serverausfall auf eine lokale Quelle ausgewichen werden.

Das **Ankündigen** bietet die Option, eine Installation anzumelden, ohne die Dateien lokal zu installieren. Ankündigungen sind vor allem für die Verteilung an mehrere Client PCs mittels Gruppenrichtlinien interessant[15].

1.2.3.2 Patches

Zum Aktualisieren von Anwendungen werden vom Windows Installer spezielle Patch Pakete zur Verfügung gestellt werden, die die Endung MSP haben. Der msiserver Dienst kann diese Dateien interpretieren, entpacken und ausführen. So kann zum Beispiel mit folgendem Aufruf ein Patch für Microsoft Office [16] eingespielt werden: *msiexec /a c:\software\office.MSI /p F:\patch.msp.*
Es wird also kein zusätzliches Programm oder Dienst benötigt, um Updates einzuspielen.

1.2.3.3 Bedeutung des Windows Installer

Durch die Anpassung mittels Transforms können Softwareinstallationen individuell durchgeführt werden. Dabei ist auch die *Installation bei Bedarf* ein nützliches Tool. Besondere Aufmerksamkeit bekommt der Windows Installer allerdings bei den Funktionen für die Gruppenrichtlinien. Durch die Gruppenrichtlinien werden Installationen im Netzwerk vereinfacht und können automatisch durchgeführt werden. Allerdings unterstützen diese nur MSI Pakte. So sind andere Paketierer auf dieses Format angewiesen, wobei hier die gängige Lösung darin besteht, existierende MSI Pakete in eigenen Installationen unterzubringen, die dann das MSI Paket aufrufen. Bei dieser Sonderform wird das Paket dann als **Wrapper-MSI** bezeichnet, da sich die Installation hier nur um das MSI „wickelt“ (s. 1.2.5.1). Oft sieht man in Installationsordnern ausführbare EXE-Dateien neben MSI Dateien, welche die MSI aufrufen und noch zusätzliche Dateien beinhalten, die meist im, von Microsoft entwickelten, CAB-Komprimierungsformat vorliegen. Diese prüfen dann, ob eine neue Version des Windows Installer vorliegt und versuchen diese zu installieren, bevor die MSI ausgeführt wird. Ebenfalls ist üblich, die MSI direkt in die EXE zu verpacken. Nützlich ist der tabellarische Aufbau der MSI, welcher ermöglicht, viele Informationen über die Installation an den Dienst zu übergeben, damit der dann fehlende

[15] Vgl.: Kapitel 1.4.3
[16] http://office.microsoft.com/de-de/

Dateien nachinstallieren kann, Upgrades verwaltet oder Custom Actions[17] nach der Installation startet, welche dann beliebigen Code ausführen können.

1.2.3.4 Fazit

Der Windows Installer ist ein mächtiges Installationsformat, besitzt aber nicht den Status des Marktführers. Für private Anwender oder kleine Betriebe sind skriptgesteuerte Installer wie das *Nullsoft Scriptable Install System* (s. 1.2.4) üblich. Für größere Unternehmen jedoch, die Wert auf netzwerkgestützte Installationen legen, führt kein Weg am MSI Format vorbei. Diese Monopolstellung soll allerdings kein Nachteil sein, denn mehrere Installer haben dieses Format in ihre Pakete integriert und bieten ebenfalls netzwerktaugliche Lösungen an.

1.2.4 Das Nullsoft Scriptable Install System (NSIS)

1.2.4.1 Arbeitsweise

Das Nullsoft Scriptable Install System der Firma *Nullsoft*[18] wird als Open Source Software angeboten[19]. Wie der Name schon vermuten lässt, handelt es sich hier nicht um eine Paketierung in Tabellen, sondern um die Installation mittels Skripten. Die Skripte können einfach über einen Texteditor verfasst werden oder über eine der zumeist quelloffenen IDEs[20]wie *Eclipse*[21] mittels Plugin (*EclipseNSIS*[22]). Damit bietet NSIS auch Linux-Kompatibilität.

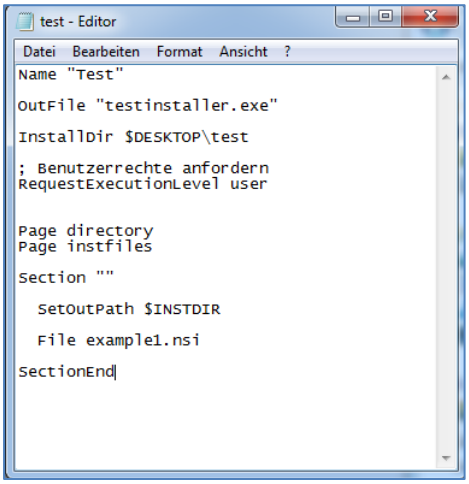

Abbildung 5: Kleines Skript zur Installation einer Datei

[17] http://msdn.microsoft.com/en-us/library/bd8h80ez(v=VS.80).aspx
[18] http://www.winamp.com/
[19] http://nsis.sourceforge.net/Download
[20] IDE: Integrated Developement Environment; Integrierte Entwicklungsumgebung
[21] http://www.eclipse.org
[22] http://nsis.sourceforge.net/EclipseNSIS_-_NSIS_plugin_for_Eclipse

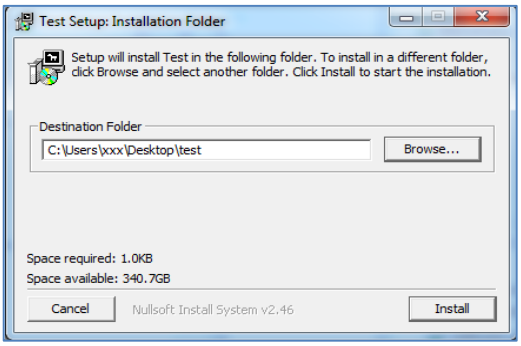

Abbildung 6: Einfache Installation mit NSIS

Die Skripte werden dann mittels eingebautem Kommandozeilen-Compiler kompiliert und die zur Installation gehörigen Dateien mittels verschiedener auswählbarer Kompressionsalgorithmen (ZLIB, BZIP2, LZMA) paketiert. Die fertigen Installationsdateien, die die Endung EXE erhalten, sind sehr schlank.

Zusätzlich werden im Internet Erweiterungen und zahlreiche Plug-Ins für NSIS angeboten, mit denen zusätzliche Funktionen aktiviert oder die Optik verändert werden können. So ermöglicht zum Beispiel das *Registry plug-in*[23] den Umgang mit Einträgen in der System Registrierung.

1.2.4.2 Fazit

Das Nullsoft Scriptable Install System findet seinen Nutzerkreis hauptsächlich bei kleineren Anwendungen für private Nutzer. Die Installationsdatei ist schnell erstellt, nimmt wenig Platz auf der Festplatte in Anspruch und ist durch die große Funktionalität durch die Plug-Ins sehr beliebt, vor allem bei der Internetgemeinde.

Durch die Inkompatibilität zu MSI sind keine Netzwerkinstallationen möglich, so dass NSIS für größere Firmen eher uninteressant ist. Durch die fehlende Nähe zum Betriebssystem können kleinere Fehler bei der Deinstallation auftreten, wie das Verbleiben von leeren Ordnern oder Registrierungseinträgen. Ebenso können nur MSI-Installationen von Windows zertifiziert werden, was für die eigentliche Zielgruppe aber eher in den Hintergrund tritt.

[23] http://nsis.sourceforge.net/Registry_plug-in

1.2.5 InstallShield

Das von Flexera Software[24] vertriebene InstallShield stellt dem Anwender eine eigene Laufzeitumgebung und ein eigenes Paketformat zur Verfügung. InstallShield bietet zudem *MSI Wrapping* (s. 1.3.2.3) an. Für die Express Version der Software ist nur eine Erstellung der MSI Pakete möglich. In den Varianten Professionell und Premier ist zudem die Konvertierung von MSI Projekten zu InstallShield Projekten oder entgegengesetzt möglich.

InstallShield bietet eine grafische Oberfläche, die bei der Erstellung von Projekten die Wahl zwischen MSI- und InstallShield-Projekten bietet. Ein sehr übersichtliches Menü führt zahlreiche Features, für die jeweils der verlinkte Inhalt der passenden Sektion in der umfangreichen Hilfe komfortabel neben dem Fenster der Einstellungen sichtbar wird (s. Abbildung 7). Zusätzlich wird ein Wizard angeboten, der Schritt für Schritt durch die Erstellung des Projekts führt.

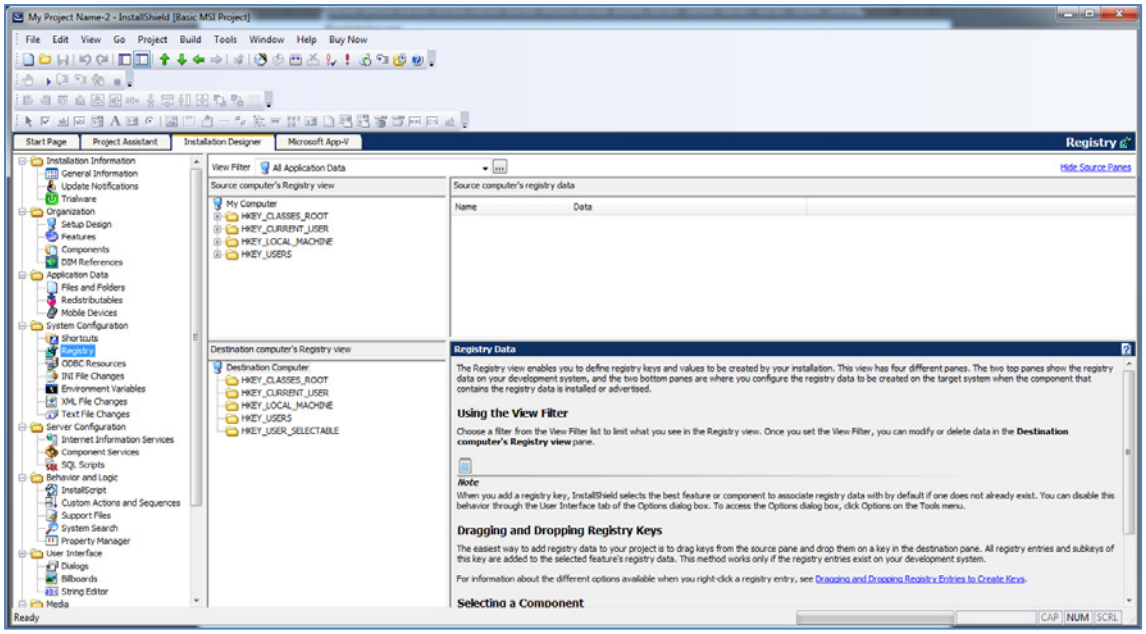

Abbildung 7: Das Menü von InstallShields ist sehr übersichtlich und bietet viele Möglichkeiten, die Installation anzupassen.

1.2.5.1 MSI, InstallScript oder beides

Bei der Erstellung eines InstallShield-Projekts erhält der User die Wahl zwischen verschiedenen Projekttypen. Die wichtigsten sind **Basic MSI Projects, InstallScript Projects und InstallScript MSI Projects**. Sie haben alle ihren speziellen Nutzen für die jeweiligen Anforderungen, sind in ihrem Nutzumfang aber auch von den Fähigkeiten des Erstellers abhängig.

[24] http://www.flexerasoftware.com/

Die Vorteile von **MSI** wurden bereits in Kapitel 1.2.2 behandelt. Durch die Nähe zum Betriebssystem steht hier die Kompatibilität zu administrativen Komponenten wie *Server Monitoring Software (SMS)* im Vordergrund. Die Verteilung von Anwendungen im Netzwerk ist nur mittels Windows Installer möglich. Nur hier können beliebige Transforms erstellt werden und auf Skripte kann man zugunsten der MSI Tabellen verzichten.

Mit **InstallScript** bedient InstallShield erweiterte Anforderungen. Mit der Skriptsprache lassen sich *Custom Actions* erzeugen, die vor und/oder nach der Installation spezielle Anweisungen ausführen wie etwa die Erstellung einer Datenbank. Auf Wunsch kann direkt in *Visual Studio*[25] gearbeitet werden. Auf die Benutzung des Windows Installers wird verzichtet. Stattdessen verwendet man die **InstallScript Engine**[26]. Sie ermöglicht verbesserten Eingriff in die Abläufe der Installation und erweitert die eher limitierte Benutzerschnittstelle von MSI Projekten. Damit besteht dann auch die Möglichkeit, in benutzerdefinierten Dialogen multimediale Elemente hinzuzufügen. Das Ergebnis der Projektkompilierung ist eine Setup.exe.

Mit dem **InstallScript MSI Project** bietet Flexera die wohl eleganteste Lösung. Sie verbindet die erstgenannten Techniken, um ihre Vorteile zu vereinen. Hierbei erhält der Windows Installer seine MSI Datei, die er ausführen kann. Gleichzeitig dient die InstallScript Engine als Benutzerschnittstelle während der Installation. Sie kann die benutzerdefinierten Laufzeit-Dialoge anzeigen und InstallScript Code ausführen.

Zur Realisierung wird die Technik des **MSI-Wrapping** genutzt. Hierbei wird das MSI Paket mit in die Setup.exe verpackt. Wird die Datei ausgeführt, öffnet die InstallScript Engine das MSI Paket. Bevor es jedoch an den Windows Installer übergeben wird, prüft die InstallScript Engine ob genügend Speicherplatz vorhanden ist. Danach lädt das Setup das kompilierte Script aus der *ISSetup.dll*. Damit kann dann auf vordefinierte Events reagiert werden und Custom Actions ausgelöst werden. Anschließend wird das MSI Paket dem Windows Installer zugewiesen. Hier können noch zusätzliche Parameter wie der Installationsmodus (zum Beispiel first-time installation, maintenance mode, patch), die Auswahl von Features und aktuelle Eigenschaftswerte übergeben werden. Nach erfolgreicher Installation werden abermals Events ausgeführt. Oft wird an dieser Stelle die installierte Applikation gestartet.

Die InstallScript Engine wirkt bei dieser Arbeitsweise **extern** auf die Installation ein, daher wird sie als **External UI Handler** bezeichnet. Bei Flexera liest man dafür die Bezeichnung *Traditional Style*.

[25] http://www.microsoft.com/germany/visualstudio/
[26] Vgl.: InstallShield Help Library, Kapitel: InstallScript Installation Projects

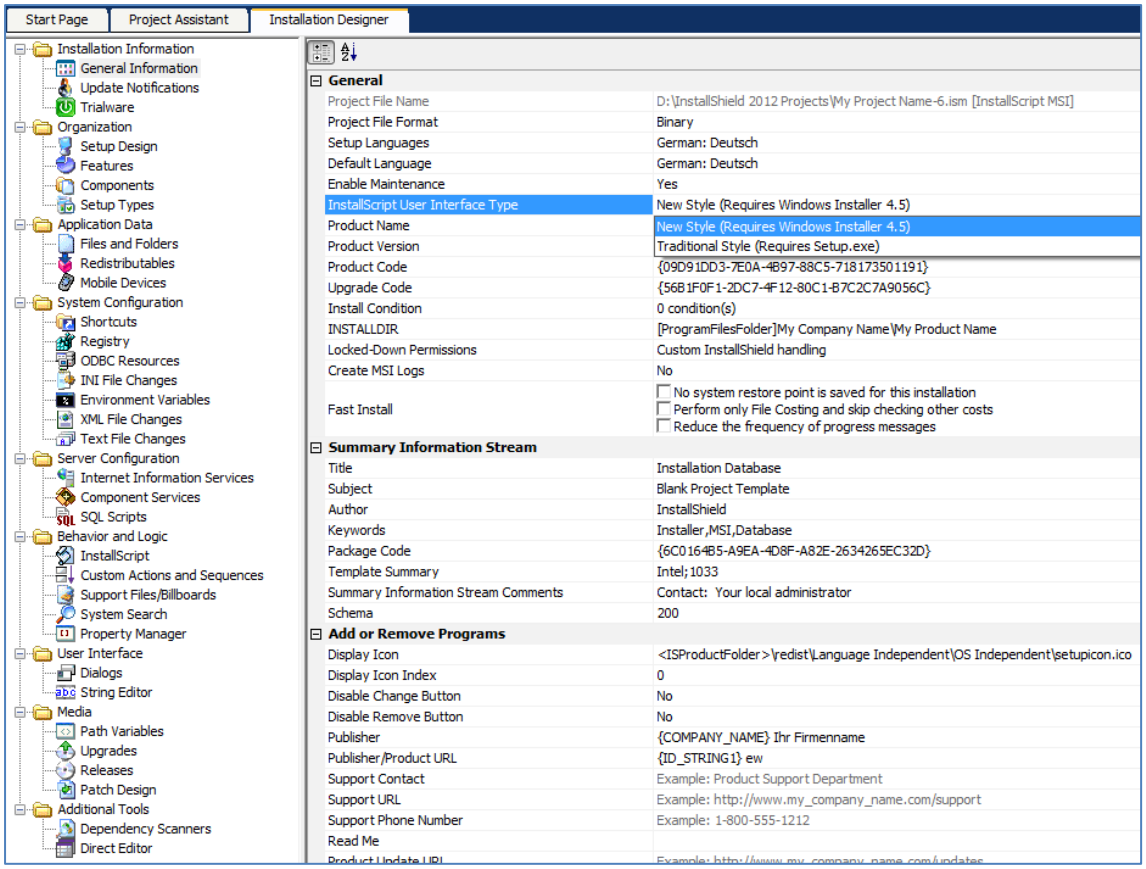

Abbildung 8: In den Projekteigenschaften kann man zwischen Traditional und New Style wählen.

Der *New Style* geht einen etwas anderen Weg. Hier arbeitet die InstallScript Engine **intern** als **Embedded UI Handler**. Intern deshalb, weil das MSI Paket direkt von der Setup.exe ausgeführt wird. Die Engine des Windows Installers extrahiert dann alle Dateien, die sich in der **MSIEmbeddedUI** Tabelle befinden und initialisiert seinerseits die InstallScript Engine, welche wiederum die ISSetup.dll lädt. Die MSIEmbeddedUI Tabelle ist ab Version 4.5 des Windows Installer verfügbar, daher muss bei älteren Versionen gepatcht oder der Old Style verwendet werden. Nach der Abarbeitung des Skripts, wie bereits beim Old Style beschrieben, übergibt die InstallScript Engine die Installation wieder an den Windows Installer.

Der Vorteil des Embedded UI Handler gegenüber dem External UI Handler besteht darin, dass für die InstallScript MSI Installation kein Setup.exe Launcher mehr benötigt wird, sondern der User direkt das MSI Paket ausführen kann. Das Paket enthält alle nötigen Informationen für die Initialisierung der InstallScript Engine. Damit erhält der User ein MSI Paket mit all seinen Vorteilen und zugleich die vielen Möglichkeiten des InstallScripts, um eine Installation perfekt auf die jeweiligen Anforderungen abzustimmen und zu individualisieren.

1.2.5.2 Funktionen

InstallShield bietet seinen Kunden eine Vielzahl an Features und Funktionen, die alle für gewisse Anforderungen nützlich sein können. Aufgrund des Umfanges beschränken wir uns auf die wesentlichen.

Die Oberfläche bietet viele Menüs, die zwar übersichtlich angeordnet sind, für Einsteiger allerdings etwas zu viel sein könnten. Dem ist sich Flexera bewusst und bietet einen **Projekt Assistenten** an, der als Wizard den User durch die Erstellung eines Projekts führt. Die wichtigsten Kenndaten und Einstellungen werden nacheinander abgefragt: der Firmenname, die Produktversion, ob ein spezielles Betriebssystem oder Software von Drittanbietern wie das .NET Framework benötigt wird, die Einbindung von Features, die zur Installation gehörigen Dateien, Shortcuts, Registry Einträge, die Installationssprache und das Quellmedium. Danach kann das Projekt direkt kompiliert werden und die Setup Datei wird erstellt.

Sollten im Lebenszyklus einer Software (s. Abbildung 1) erweiterte Anforderungen an eine Installation bestehen und im Projekt etwa die Feinheiten des InstallScripts verwendet werden, besteht die Möglichkeit, es von einem reinen MSI-Projekt in ein InstallScript MSI-Project umzuwandeln. Dabei entfallen dann allerdings die bisherigen Windows Installer Dialoge, die durch neue InstallScript Dialoge ersetzt werden müssen. Der Weg zurück zum Basic MSI Project ist auf die Lizenz der Premier Version von InstallShield beschränkt. In der Professional Edition ist nur noch die **Konvertierung** vom InstallScript MSI-Project zum InstallScript Project möglich. Für alle anderen Projektkonvertierungen (s. Abbildung 9) wird die Premier Edition benötigt. Zusätzlich können Visual Studio Setup Projects in Basic MSI Projects importiert werden.

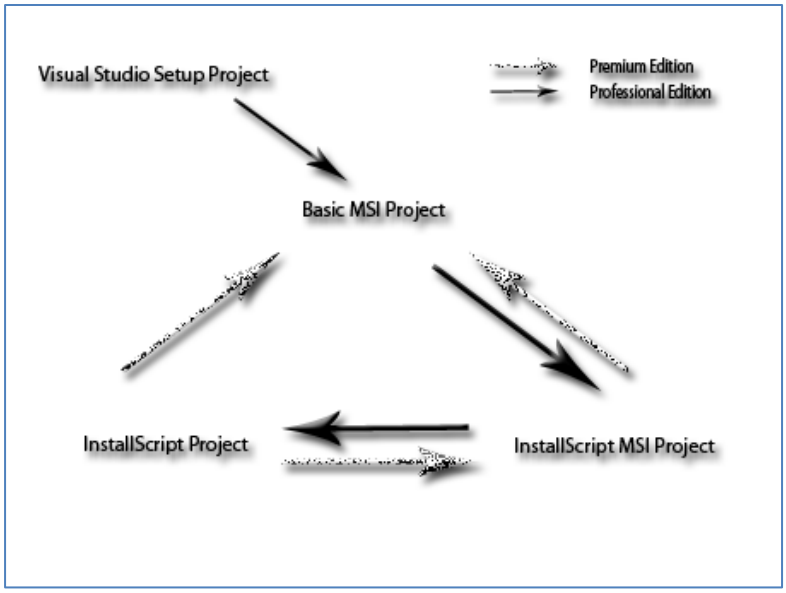

Abbildung 9: Für einige Projektkonvertierungen wird die teure Premium Edition benötigt.

Der Import von Windows Installer Modulen ist ebenfalls möglich. Dieser Vorgang wird **Merging Modules**, zu Deutsch „Module verschmelzen“, genannt. Diese Merge Module sind Windows Installer Komponenten, die Setup Logik enthalten und dazu dienen, Code, Dateien, Ressourcen und Registry Einträge weiterzugeben.[27] InstallShield führt diese Operation über einen Wizard durch, der automatisch Fehler beim Import, wie etwa Redundanzen, erkennt.

InstallShield bietet die Möglichkeit, **Transforms** anzulegen oder zu importieren. Hierzu steht ebenfalls ein Wizard zur Verfügung. Die Transforms besitzen Windows Installer Standard und damit die Endung MST.

Die Premier Edition beherbergt noch ein für die kommerzielle Nutzung interessantes Feature: den **InstallShield Activation Service**. Dabei handelt es sich um einen gebührenpflichten Service, der Unternehmen die Produktaktivierung und Lizenzverwaltung vereinfacht. Gebührenpflichtig ist der Dienst deshalb, weil er sich auf Servern von Flexera befindet. Dort werden die Lizenzen gespeichert und können vom eigenen Unternehmen verwaltet werden.

Zur Virtualisierung enthält InstallShield ein **Virtualization Pack**.[28] Das beinhaltet den *Microsoft App-V Assistant.* Damit können Anwendungen auf den Computern der Endanwender bereitgestellt werden, ohne sie direkt auf deren Rechnern zu installieren.

1.2.5.3 Update

Neben dem **Full Release** bietet InstallShield Möglichkeiten zum **Patching** und zum **Differential Release**. Letzteres trifft auf InstallScript Projekte zu. Das Differential Release enthält Dateien, die zur älteren Version veränderte, also differenzierte, Dateiinformationen besitzen. Hat eine Datei andere Attribute (wie z.B. Datum, Zeit oder Größe) als die auf dem Zielrechner befindlichen, ersetzt sie diese. Ähnlich arbeitet das Full Release. Hier werden allerdings nicht Dateien auf geänderte Daten oder Attribute untersucht, sondern es wird ein komplettes Release übertragen, welches in Abhängigkeit von einer Versionsänderung, also *version specific*, oder standardmäßig, als non-version specific, alle vorhandenen Dateien überschreibt und neue installiert. Wird die versionsunabhängige Variante gewählt, kann auch eine Vollinstallation ohne vorherige Installation eingespielt werden. Das Full Release wird bei InstallScript Projekten eingesetzt.

[27] http://msdn.microsoft.com/en-us/library/windows/desktop/aa369820(v=vs.85).aspx
[28] http://technet.microsoft.com/de-de/library/cc843848

Beim Patching, welches bei InstallShield eine auf MSI Projekte und InstallScript MSI Projekte beschränkte Bedeutung hat, wird eine frühere Version des Windows Installer Pakets erneuert, welches dann die Applikation auf den neuesten Stand bringt.

Für kleinere Updates kann ein **Quickpatch** Verfahren eingesetzt werden. Es kann entweder auf vorhandene Basic MSI Projects oder InstallScript MSI Projects angewendet oder als separates Quickpatch Project erstellt werden.

1.2.5.4 Bedeutung InstallShield

InstallShield bietet das „Rundum-Sorglos-Paket". Die Übersichtlichkeit sämtlicher Funktionen ist trotz des großen Umfangs gegeben. Für Einsteiger bieten sich der Assistent sowie die umfangreiche Hilfe an.

Die Verbindung von Windows Installer gesteuerten Installationen mit gescripteten Routinen vereint die Vorteile beider Systeme. Dank der MSI Einbindung muss nicht auf das begehrte Windows Logo verzichtet werden und die Kompatibilität mit der Windows Umgebung ist gegeben. Software kann damit im Netzwerk verteilt werden sowie mittels der App-V Komponente virtualisiert werden.

1.2.5.5 Fazit

InstallShield darf sich seit Jahren als Marktführer unter den Herstellern von Installationsroutinen bezeichnen. Microsoft empfiehlt InstallShield als das Programm der Wahl für die Erstellung von Installationen für Visual Studio-Anwendungen. Zahlreiche große Software- und Hardwarehersteller wie Adobe, Sun, Dell oder Fujitsu vertrauen auf das Programm.[29] Durch den gehobenen Preis, vor allem der Premier Edition, ist InstallShield vor allem für Firmen mit einem größeren Kundenkreis interessant.

[29] Vgl.: http://www.flexerasoftware.com/products/installshield.htm

1.2.6 Bewertung Installationsverfahren

Die Beschreibung von Softwareinstallationen mittels Tabellen als Unattended Setup ist eine sehr elegante Methode für Installationsverfahren. Man macht sich vorgefertigte Installationen zu Eigen und verändert diese nach Wunsch.

Der Windows Installer verfügt über eine Nähe zum Betriebssystem, die viele Features im Zusammenhang der Softwareinstallation möglich macht. Wichtig ist dabei die Verteilung im Netzwerk, die nur mittels MSI möglich ist.

Scriptverfahren wie beim Nullsoft Scriptable Install System sind hingegen flexibler, da spezielle Einstellungsmöglichkeiten eingebunden werden können, die nicht im Umfang von MSI enthalten sind. Das ist jedoch meist mit einem erhöhten Arbeitsaufwand verbunden.

Die Zusammenführung von MSI und Skript bei InstallShield ist wohl die mächtigste und zugleich flexibelste Variante der Installation. Die Vorteile werden kombiniert und Installationen lassen sich frei nach Wunsch erzeugen.

Weitere Installer sind zum Beispiel InstallAware[30], Inno Setup[31] oder InstallBuilder[32]. Ihre Existenz ist durchaus berechtigt, denn sie bedienen Anforderungen je nach Ansprüchen, jedoch soll hier aus Gründen des Umfangs nur darauf verwiesen werden.

1.3 Softwareverteilung

In Firmennetzwerken mit einer größeren Anzahl an End-User PCs ist es heutzutage üblich, den einzelnen Usern nicht die volle Gewalt über ihre Rechner zu übertragen. Meist fehlen zumindest die Administratorprivilegien, oft aber auch weitergehende Rechte, wie die der Installation von Software. Hier kommt die Softwareverteilung ins Spiel. Dem Administrator sind einige Mittel gegeben, um die gewünschte Software auf den PCs zu verteilen. Diese sollen in diesem Kapitel behandelt werden.

[30] http://www.installaware.de/
[31] http://www.jrsoftware.org/isinfo.php
[32] http://installbuilder.bitrock.com/

1.3.1 Prinzipien der Softwareverteilung

Der Verteilung von Software durch Administratoren liegen gewisse Prinzipien zugrunde. Diese werden folgend erläutert.

1.3.1.1 Das Push Prinzip

Beim Pushing werden die zu installierenden Programme vom Administrator an den End-User PC gesendet. Dort können Sie dann, oft ohne GUI, installiert werden. Dadurch hat der Benutzer keine Eingriffsmöglichkeit in die Installation.

Diese Vorgehensweise ermöglicht einen einheitlichen Stand der Software an sämtlichen Client-PCs. Optional kann auch Software für spezielle Benutzer oder Benutzergruppen installiert werden, für andere nicht.

1.3.1.2 Das Pull Prinzip

Im Gegensatz zum Push Prinzip wird beim Pulling Software nicht automatisch eingespielt, sondern nur bereitgestellt. Den Usern wird dann gemeldet, dass Software zur Installation verfügbar ist. Sie können sie dann vom zentralen Punkt aus lokal installieren.

Der Stand der Software ist beim Pulling schwer zu vereinheitlichen, jedoch kann der Nutzer, im Unterschied zum Pushing, meist zwischen mehreren zu installierenden Programmen wählen.

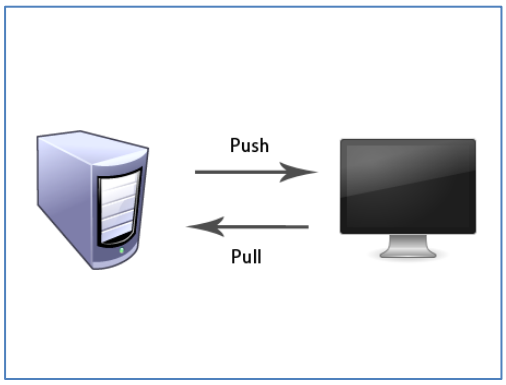

Abbildung 10: Push und Pull Prinzipien

1.3.1.3 Fat-und Thin-Clients

Eine weitere Möglichkeit der Softwareverteilung stellen die Fat- und Thin-Clients dar. Dabei werden Grafische Oberfläche, Anwendungslogik und Daten nach Bedarf voneinander getrennt. So wird die Software nur teilweise verteilt.

Der **Thin-Client** übernimmt nur die Aufgaben der Präsentationsschicht. Er greift auf die Software zu, die zentral installiert ist und stellt diese auf dem User-PC dar. Das erspart einzelne Installationen und nicht zuletzt auch Platz auf den PCs. Dadurch entstehen allerdings höhere Anforderungen an den Server und das Netzwerk. Greifen mehrere Rechner auf die zentralisierten Programme zu, vervielfacht sich der Rechenaufwand für das System und die Netzwerkauslastung.

Beim **Fat-Client** wird die Anwendungslogik vom Server auf die User-PCs ausgelagert. Lediglich die Daten verbleiben auf dem Server. Hierbei entsteht zwar etwas weniger Traffic für das Netzwerk, die Beanspruchung der CPU des Servers sinkt allerdings markant. Der Nachteil ist, dass nun auf jedem Rechner eine Installation, am besten nach Push- oder Pull-Prinzip, durchgeführt werden muss.

Der Vorteil von Fat- und Thin-Client Lösungen ist die Zentralisierung von mindestens der Datenschicht von Softwarekomponenten. Dadurch kann Dateninkonsistenz vermieden werden und Benutzer erhalten Zugriff auf die Gemeinschaftsdaten, wenn sie die nötigen Rechte besitzen. Das erleichtert die Zusammenarbeit an Projekten. Wird, wie beim Thin-Client, zusätzlich die Programmlogik zentralisiert, ist zudem der Stand der Programme vereinheitlicht. Durch die Installation an nur einem Punkt im Netzwerk wird nur ein Update für genau diesen Punkt benötigt.

Außer Fat- und Thin-Client sind noch **Rich- und Smart-Client** übliche Verfahren.[33] **Rich-Clients** sind eigentlich Thin-Clients, also Teile der Präsentationsschicht, die jedoch komplexere Operationen ausführen müssen, als die pure Anzeige der Programme. Das liegt oft an der Komplexität der Präsentationsschicht, wie zum Beispiel bei der Anzeige von Videos, für die auf dem Client PC zunächst Decodierungen stattfinden müssen, bevor sie abgespielt werden können. Somit hat der Client entsprechend mehr zu leisten und kann nicht mehr als Thin-Client bezeichnet werden.

[33] Vgl.: Chantelau / Brothuhn, Multimediale Client-Server-Systeme, 2009, S. 41

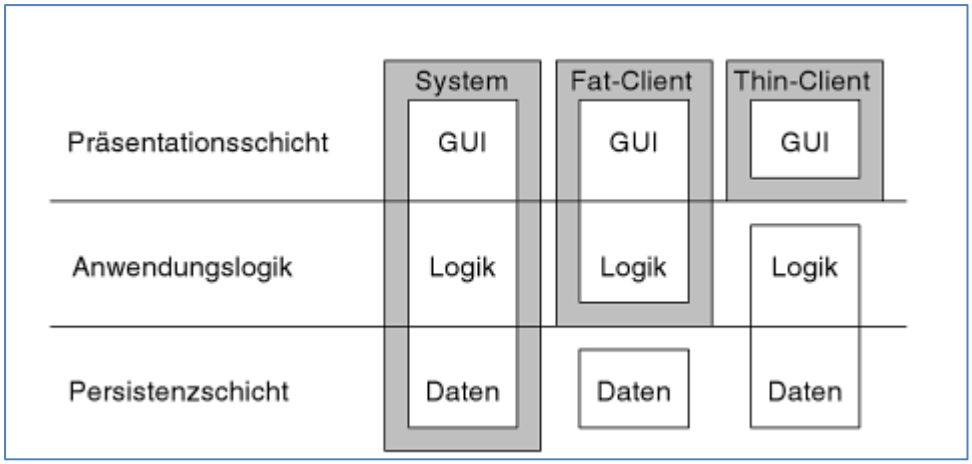

Abbildung 11: Fat-und Thin-Clients im Schichtmodell

Smart-Clients sind Rich- oder Fat-Clients, bei denen vor oder während der Benutzung des jeweiligen Programms auf dem Nutzer-PC die Anwendung vom Server aus automatisch installiert oder aktualisiert wird. Hier greift also das Push-Prinzip. Der Nutzer bekommt im Normalfall gar nichts von dem Eingriff mit.

Erfolgt keine Aufteilung von Programm-, Daten- und Präsentationslogik, spricht man von einem **Monolithischen System**. Dabei werden die Anwendungen komplett auf allen Client-PCs installiert und arbeiten unabhängig voneinander.

1.3.2 Verteilung mittels Netzwerkfreigabe

Der einfachste Weg, Software im Netzwerk zu verteilen, ist die Netzwerkfreigabe. Die Software wird auf dem Server freigegeben und kann von jedem Nutzer auf den Client-PC kopiert und dann per Hand oder mittels Skript installiert werden. Dabei kann auf spezielle Software auf Kosten von Konfigurationsmöglichkeiten verzichtet werden. Außerdem hat der Administrator damit kaum Einfluss auf den Zeitpunkt und den Ort der Installation. Er muss im Zweifelsfall an jeden Rechner gehen und die Software manuell installieren. Dieses Vorgehen ist für Updates denkbar ungeeignet, da nur unter erheblichen Schwierigkeiten für einen einheitlichen Stand im Netzwerk gesorgt werden kann.

1.3.3 Gruppenrichtlinien im Active Directory-Netzwerk

1.3.3.1 Active Directory

Beim *Active Directory*, also *aktives Verzeichnis,* handelt es sich um ein Verzeichnis, welches Informationen über Benutzer, Benutzergruppen und Computer enthält. Außerdem können Informationen zur Konfiguration von Anwendungen wie dem *Exchange Server* oder dem *Internet Security and Acceleration Server* bzw. dessen Nachfolger, der *Forefront Threat Management Gateway* gesammelt werden.[34] Mittels eines Verzeichnisdienstes können Berechtigungen und Freigaben verteilt und Anmeldungen an PCs verwaltet werden. Die Verzeichnisse sind meist hierarchisch aufgebaut und durch das Active Directory durch verschiedene Standards definiert. Damit können gezielt Informationen gesucht, angelegt, verändert, gelöscht und angewendet werden. Diese Aktionen sind durch Berechtigungen geregelt.

1.3.3.2 Gruppenrichtlinien

Zur Verwaltung von Benutzer- und Computereinstellungen in einem Active Directory werden *Gruppenrichtlinien* verwendet. Damit können zahlreiche Einstellungen automatisch vorgenommen werden, wie zum Beispiel das Verhalten des Browsers oder die Definition von Kennwörtern. Sie arbeiten mit speziellen Registry-Schlüsseln, die nicht permanent sind, das heißt, sie enthalten nur solange die nötigen Informationen, wie das Gruppenrichtlinienobjekt (GPO), welches sie definiert, gültig ist. Dazu werden die Gruppenrichtlinien in Abständen aktualisiert. Um unberechtigte Zugriffe zu vermeiden, sollte nur der Systemadministrator über Berechtigungen verfügen.[35]

[34] Vgl.: Joos, Planungsbuch Microsoft-Netzwerke, 2006, S. 73
[35] Vgl.: Schwichtenberg, Windows Vista Business, 2008, S. 590

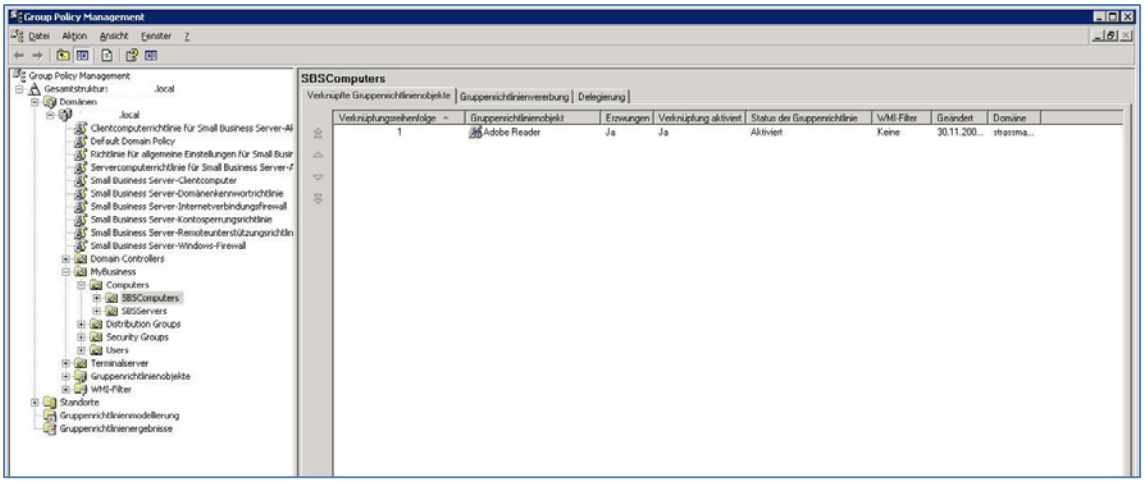

Abbildung 12: Hier wurde der Adobe Reader zur Computer-bezogenen Verteilung eingerichtet.

Mittels Gruppenrichtlinien ist es möglich, ein Softwarepaket automatisch an alle Client-PCs zu übertragen. Dafür muss die Software als MSI vorliegen[36]. Liegt kein MSI Paket vor, muss die Software repaketiert werden oder ein Wrapper MSI genutzt werden.[37] Bei der Softwareverteilung durch Gruppenrichtlinien kann die Software entweder für Computer oder Benutzer bereitgestellt werden. So kann etwa eine Office Anwendung für alle Benutzer eines Computers installiert werden oder eine speziellere Anwendung wie ein CAD Programm nur für einen bestimmten Benutzer. Dadurch wird auch der Zeitpunkt der Installation bestimmt. Bei der Verteilung pro Computer wird die Software bereits beim Systemstart installiert, also zum Zeitpunkt der Übernahme der Richtlinie. Wird hingegen benutzerbezogen installiert, erfolgt die Verteilung erst bei der Anmeldung des richtigen Benutzers. Startet der Benutzer die Anwendung, ist die Installation abgeschlossen.

1.3.3.3 Anwendung

Für die Verteilung muss das MSI Paket mit allen Ordnern im Netzwerk freigegeben werden. Software, die für die Verteilung mittels Gruppenrichtlinien ausgelegt ist, verfügt hier meist über Automatismen, die diese Aufgabe erledigen. So kann man zum Beispiel bei der Installation von Microsoft Office die Installationsdatei mit einem Parameter zur administrativen Installation (setup.exe /a) starten, worauf sie die Daten korrekt zum Verteilungspunkt hin installiert.[38] Alle Nutzer sollten Leserechte auf das Verzeichnis erhalten. Darüber hinaus benötigen bei der Verteilung pro Computer auch die Computer Leserechte.

[36] Vgl.: Westbrook ,Active Directory für Windows Server 2008,2009, S.580
[37] Vgl. Kapitel 1.3.5.1
[38] Vgl.: http://support.microsoft.com/kb/314934/de

Die Verteilung der Software kann nur via *Zuweisen* oder *Veröffentlichen* vorgenommen werden. Beim Zuweisen wird die Installation am Client-PC entweder beim Systemstart oder bei der Benutzeranmeldung (siehe Kapitel 1.3.3.2) ausgeführt. Dies entspricht dem Push-Prinzip[39]. Wird die Software veröffentlicht, wird sie nicht direkt installiert, sondern steht dem Benutzer in der Systemsteuerung je nach Betriebssystem unter *Software* bzw. *Programme und Funktionen* zur Installation bereit. Außerdem kann noch der Aufruf einer mit der Software verknüpften Datei zur Aufforderung der Installation führen. Das Veröffentlichen entspricht dem Pull-Prinzip[40] und kann nur an Benutzer, nicht an Computer erfolgen.

Möchte man ein Softwarepaket aktualisieren, besteht die Möglichkeit, eine Gruppenrichtlinie erneut bereitzustellen. Dabei wird die Anwendung auf allen Computern erneut installiert, auf denen sie bereits installiert ist.

1.3.3.4 Fazit

Die Verteilung von Software über Gruppenrichtlinien ist einfach einzurichten und zu verwalten. Sie ist für kleinere und mittlere Netzwerke völlig ausreichend. Die Voraussetzung von MSI und Active Directory sowie weitere Konfigurationswünsche machen trotz ständiger Erweiterung der Möglichkeiten der Gruppenrichtlinien für größere Unternehmen spezielle Verteilungssoftware interessanter. Zum Update ist diese Form untauglich, da die Anwendungen immer nur neu installiert werden können.

1.4 Softwaremanagement

Die Vereinigung mehrerer Bereiche der automatisierten Verwaltung von Rechnernetzen wird als Softwaremanagement bezeichnet. Die wichtigsten Aufgaben von Softwaremanagementlösungen sind die automatisierte Betriebssysteminstallation, die automatisierte Verteilung und Installation von Anwendungen im Netzwerk, automatische Verteilung von Patches und die Katalogisierung der Computer, oft inklusive deren Fernsteuerung bzw. Fernwartung.

[39] Vgl.: Kapitel 1.4.1.1
[40] Vgl.: Kapitel 1.4.1.2

Sie bestehen generell aus 2 Komponenten: einem Server(-dienst) und spezieller Client-Software, die auf den Rechnern im Netzwerk installiert wird. Der Server steuert die Verteilung der Pakete und enthält Informationen über die Clients in einer Datenbank. Die Software der Clients wird oft schon bei der Installation des Betriebssystems mitinstalliert. Sie ist zuständig für die Installation der Anwendungen und überträgt Informationen an den Server.

1.4.1 Microsoft System Center Configuration Manager

Der *Microsoft System Center Configuration Manager (SCCM)* ist die Softwaremanagementlösung von Microsoft. Die wichtigsten Funktionen des SCCM sind *Betriebssystembereitstellung*, *Softwareverteilung*, *Softwareupdateverteilung*, *Software Asset Intelligence* und *Desired Configuration Management*.[41]

1.4.1.1 Funktionen

Mit der **Betriebssystembereitstellung** können Betriebssysteme aus der Ferne installiert und konfiguriert werden. Das ist durch die Kopplung des SCCM mit *Windows Deployment Services* möglich.[42]

SCCM vereinfacht die **Softwareverteilung** für Benutzergruppen ähnlich wie bei der Gruppenrichtlinienverteilung. Es beherrscht unter anderem Ankündigen, Zeitpläne zur Installation und Remoteaktivierung (Wake On LAN).[43] Außerdem ist es möglich, in einem Paket mehrere Anwendungen unterzubringen, die getrennt voneinander installiert werden können. Die Pakete werden dann mit dem jeweiligen Parameter für die zu installierende Software angesprochen.

Die auf *Microsoft Windows Software Update Services (WSUS)* basierende **Softwareupdateverteilung** hingegen kann Updates für Produkte von Microsoft und Drittanbietern sowie Hardwaretreiber und das System-BIOS bereitstellen. Verbinden sich Clients nur selten mit dem Firmennetzwerk, können Updates via *Internet Based Client Management (IBCM)* über das Internet bezogen werden. Zusammen mit *Virtual Machine Manager* können auch virtuelle Maschinen und die darauf installierte Software auf dem neuesten Stand gehalten

[41] Vgl.: http://www.microsoft.com/de-de/server/system-center/2007-2010/configuration-manager/default.aspx
[42] Vgl.: Panek/Wentworth, Mastering Microsoft Windows 7 Administration, 2010, S, 54
[43] Vgl.: http://www.microsoft.com/de-de/server/system-center/2007-2010/configuration-manager/funktionen.aspx

werden, auch wenn die virtuellen Maschinen gar nicht ausgeführt werden. Damit der Traffic am zentralen Server entlastet wird, können Verteilungsstationen an verschiedenen Standorten eingerichtet werden.

Asset Intelligence bietet Systemadministratoren einen Überblick über den Hard- und Softwarebestand innerhalb eines Unternehmens. Es liefert Auskunft, wer diesen nutzt und von welchem Ort. Dazu werden zahlreiche Detailinformationen gesammelt und katalogisiert, was dem Administrator etliche Statistiken und Möglichkeiten zur frühen Fehlererkennung liefert.

Als **Desired Configuration Management** bezeichnet Microsoft die Prüfung der Clients. Dabei wird geprüft, ob die Clients korrekt nach Vorgaben konfiguriert sind. Es würde zum Beispiel eine falsche Betriebssystemversion oder falsche Update- oder Sicherheitseinstellungen melden.

1.4.1.2 Fazit

Der Microsoft System Center Manager eignet sich gut für große Netzwerke, die auch in Subnetze unterteilt sein können. Das Netzwerk eines internationalen Unternehmens kann von einem Ort aus gesteuert und mit Software versorgt werden. Auch die Verwaltung der Software ist möglich. Aufgrund der Komplexität ist es jedoch für kleinere Unternehmen nicht empfehlenswert, dort reicht meist die Installation über Gruppenrichtlinien aus. Die Möglichkeit, Updates auch ohne Firmennetzwerkzugang über das Internet zu beziehen, ohne dass der Nutzer sich selbst darum kümmern muss ist ein gelungenes Feature, welches die Aktualität von Anwendungen und damit ihre Sicherheit steigert.

1.4.2 Open PC Server Integration

Open PC Server Integration (Opsi) der deutschen *uib GmbH*[44] ist ein Open Source Softwaremanagementsystem, das Linux Server zur Verwaltung von Windows-Clients benutzt. Opsi bietet genauso wie SCCM *Betriebssystembereitstellung*, *Softwareverteilung*, *Patch-Management* und *Inventarisierung*.

[44] Vgl.: http://www.opsi.org

1.4.2.1 Funktionen

Als Kern der **Softwareverteilung** arbeitet ein Linux Server. Er wird hier als **Depot** bezeichnet. Ähnlich wie beim SCCM können in größeren Netzwerken weitere Depots erstellt werden, denen Clients zugeordnet werden. Sollte sich ein Client allerdings nicht in der Nähe seines Heimatdepots befinden, etwa weil er zwischen verschiedenen Filialen wechselt, kann er Gebrauch von der dynamischen Depotzuweisung machen. Dieses Feature erlaubt ihm, Software von Nicht-Heimat-Depots im WAN zu beziehen. Dazu ist neben der Unattended Installation und der Zuweisung von Software (Push-Betrieb) durch den Admin auch ein **Software-on-Demand** (Pull-Betrieb) zum Umfang des Tools hinzugekommen. Mit dieser Technologie ist auch ein Einsatz in der Cloud möglich, jedoch sollten hier zusätzliche Sicherheitsmaßnahmen wie VPNs getroffen werden.

Neben der Verteilung bietet Opsi auch Möglichkeiten zum **Patchen** von Software. Dazu sammelt der Administrator Informationen (Inventarisierung) der Clients und weist ihnen Updates zu oder bietet sie per On-Demand an.

Außerdem kann Opsi Lizenzen verwalten. Es sammelt Informationen darüber, welcher Client über welche Lizenzen verfügt und wie viele noch frei sind.[45]

1.4.2.2 Fazit

Opsi ist Open Source und kostet trotzdem. Denn nur die Kernfunktionen SCCM *Betriebssystembereitstellung*, *Softwareverteilung*, *Patch-Management* und *Inventarisierung* sind kostenfrei. Die WAN Erweiterung, Software-on-Demand, Lizenzmanagement und weitere Features wie Wake-On-Lan sind als kostenpflichtige Kofinanzierungsprojekte ausgewiesen. Sie werden erst kostenfrei, wenn die Entwicklungskosten eingespielt wurden.

An die Funktionalität eines Microsoft System Center Configuration Managers kommt Opsi nicht heran. Dafür ist es im Kern kostenfrei und bietet Einsatzmöglichkeiten, die für mittelständige Unternehmen ausreichen sollten. Bisher ist die Verbreitung von Opsi weitgehend auf den deutschsprachigen Raum beschränkt.

[45] http://www.admin-magazin.de/content/download/3071/30194/fileneuen-Opsi-4.0.1.pdf

1.4.3 FrontRange HEAT

Die kalifornische Firma FrontRange[46] ist der Marktführer für Produkte des Software Lifecycle Managements.[47] **Die HEAT Client Management** Software wurde maßgeblich beim Kauf der deutschen Firma *enteo* im Jahr 2007 mit ihrem Flaggschiff *NetInstall* zusammen mit dem hauseigenen Produkt zum Management von Diensten und Infrastrukturen zur *HEAT Service Management Platform* ins Unternehmen eingebracht. [48]

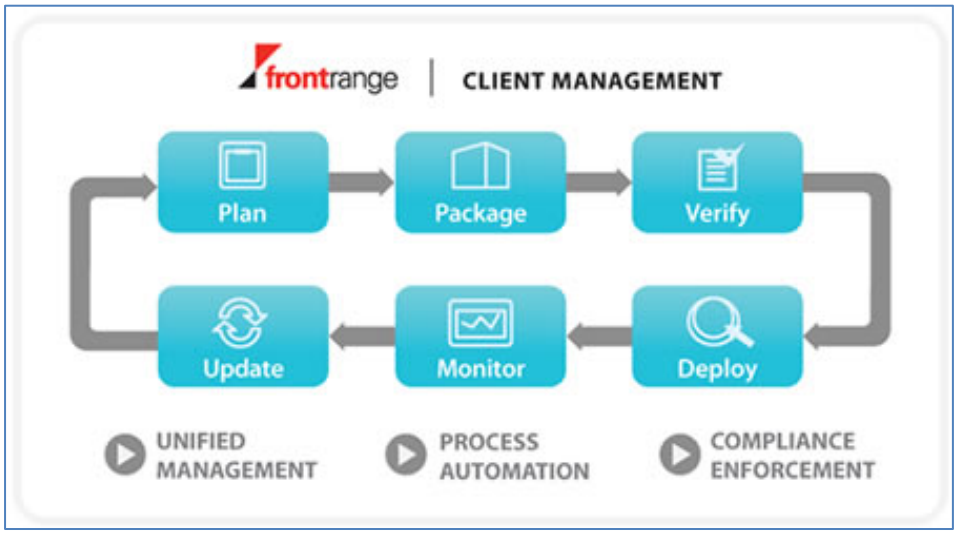

Abbildung 13: Heat Client Management Lebenszyklus

Die HEAT Client Management Software befähigt den Administrator wie bei den zuvor behandelten Tools[49] zur *Bereitstellung von Betriebssystemen*, *Softwareverteilung*, *Patch Management* sowie *Inventarisierung*.

1.4.3.1 Funktionen

HEAT **Betriebssysteminstallationen** laufen weitgehend automatisch ab, wobei hier im Speziellen automatische Backups von laufenden Systemen gemacht werden können. Diese können dann auf andere Clients migriert werden, was viel Zeit spart, die sonst in Treiberinstallationen und Konfigurationen investiert werden müsste. HEAT arbeitet im Gegensatz zu Opsi und SCCM mit plattformunabhängigen Clients.

[46] Vgl.: http://www.frontrange.com
[47] Vgl.: http://www.msg-systems.com/sysserv_sw_lifecycle.0.html
[48] Vgl.: http://www.heise.de/netze/meldung/Frontrange-kauft-Enteo-155747.html
[49] Vgl.: Kapitel 1.5.1 (Microsoft System Center Configuration Manager); Kapitel 1.5.2 Open PC Server Integration

Die **Softwareverteilung** wird mit Wizards erleichtert - die Installation, Konfiguration und laut Hersteller das Testing[50] von Anwendungen. Auch können diverse Programme selbst paketiert und anschließend verteilt werden.

HEAT sucht selbständig nach neuen **Patches** und kann sie je nach Notwendigkeit sofort oder regelbasiert verteilen.

Die **Inventarisierung** sammelt nicht nur Informationen über die Client PCs sondern über alle Geräte, die eine IP-Adresse haben, wie Drucker, Smartphones oder Tablets. Damit können für alle Geräte Berichte erstellt werden, die für eine Qualitätssicherung sorgen.

Zusätzliche Features sind neben anderen *Remote Control*, *Mobile Device Management* für mobile Endgeräte, Lizenzmanagement und das automatische Backup und Migrationstool *Personality Migration & Disaster Recovery*, das im Falle von Abstürzen oder sonstigen Problemen für eine schnelle Systemwiederherstellung sorgen kann. Ebenso werden mehrere Server im WAN unterstützt, die den Traffic gleichmäßig über mehrere Stationen verteilen sollen. Die zahlreichen weiteren Funktionen aufzuzählen, würde den Rahmen sprengen. Gewiss ist, dass die mächtige Funktionalität nicht auf Kosten der Handhabung zu kaufen ist. Denn komfortabel ist HEAT allemal.

1.4.3.2 Fazit

HEAT ist zu Recht Marktführer bei den Softwaremanagementlösungen. Der große funktionelle Umfang sowie die leichte Handhabung durch Wizards sorgen für ein stimmiges Gesamtpaket. So ist es ein Tool, welches für kleinere, mittlere und auch große Firmen in Frage kommt und auch eingesetzt wird.

1.4.4 ZENworks

ZENworks (Zero Effort Networks for users), die Softwaremanagementlösung der Firma Novell[51], bietet *Anwendungsvirtualisierung*, *Configuration Management* und *Endpoint Security*. Für uns am interessantesten ist das **Configuration Management**, das für die Softwareverteilung und –anwendung sowie die Implementierung und Verwaltung zuständig ist. ZENworks verwaltet Rechner im Netzwerk mittels Nutzung des *NDS (Novell Directory Services)*, den von Novell

[50] Vgl.: http://www.frontrange.com/client-management/
[51] Vgl.: http://www.novell.com

entwickelten Verzeichnisdienst, der heute zusammen mit *LDAP*[52] Funktionen als *eDirectory* bezeichnet wird.[53]

Das Configuration Management beherrscht für die Softwareverteilung Techniken zur Bündelung und Implementierung von Software, Imaging und Preboot-Services, kann Clients mehrere Verteilungspunkte zuweisen und liegt einer rollenbasierten Verwaltung zugrunde, so dass genaue Einstellungen für Rechte der Nutzer und Administratoren im Netzwerk angegeben werden können.

1.5 Bedeutung von Softwareverteilung und Softwaremanagement für Update Verfahren

In den vorhergehenden Kapiteln haben wir uns vor allem mit der Installation von Software beschäftigt und verschiedene Probleme und Lösungsmöglichkeiten aufgezeigt. Doch was bedeutet das für die Instandhaltung? Zunächst muss hierbei betrachtet werden, wie die Software vorliegt, die installiert und gewartet werden soll. Dann kann man sich überlegen, wie die Instandhaltung durchgeführt werden soll. Handelt es sich um ein kleines Tool, dessen Dateien bei neueren Versionen komplett überschrieben werden, ist die Weitergabe über Netzwerkfreigabe oft ausreichend. Sollen dazu noch Einstellungen getätigt oder silent gepatcht werden, kommen Gruppenrichtlinien ins Spiel. Allerdings ist Patching hier immer mit Aufwand des Administrators verbunden, der alte Dateien im Verteilungsverzeichnis durch neue ersetzen und je nach Art der Verteilung an die Rechner der Clients muss oder ihnen neue Software ankündigen bzw. einspielen muss. Verfügt die Software über ein eigenes Patchtool und kann sie Aktualisierungen über das Internet beziehen, muss der Client über eine Internetverbindung verfügen. Der Admin braucht sich dann nur um die Installation kümmern. Hier kann dann aber das Problem entstehen, dass verschiedene Clients über verschiedene Versionen der Software verfügen, was bei Gruppenarbeiten zu Problemen führen kann. Bei solchen Anforderungen sind die Softwaremanagementlösungen besser aufgestellt. Sie können die Software auf jedem Client-PC immer auf einem einheitlichen Stand halten, da sie Software protokollieren, einspielen und aktuell halten können. Allerdings spielt hier der Kostenfaktor oft eine wichtige Rolle, denn die professionellen Manager lassen sich ihre Funktionalität und den Komfort gut bezahlen. Hier gilt es abzuwägen, welche Lösung für das eigene Netzwerk geeignet ist.

[52] LDAP : Lightweight Directory Access Protocol: Anwendungsprotokoll zur Kommunikation zwischen Client und Verzeichnisdienst
[53] Vgl.: Larisch, Netzwerktechnik, 2005, S.292

Die Bedeutung von Systemen zur Verteilung von Software kann objektiv nur an den jeweiligen Einsatzgebieten sowie deren Anforderungen für die Softwareverteilung gemessen werden. Kleine und mittelständige Unternehmen haben selbstverständlich andere Anforderungen an gewisse Qualitätsmerkmale als große Konzerne und gewichten einige Features anders, weil sie im konkreten Fall eine größere Bedeutung besitzen. Das trifft natürlich nicht nur auf die Größe eines Unternehmens sondern auch auf spezielle Bedürfnisse, Marktanteile, die Ausrichtung sowie natürlich die gesamte IT-Infrastruktur zu; Faktoren, die maßgeblich die Anforderungen an Update Systeme bestimmen. Oft besteht die Infrastruktur aus mehreren Sub-Netzwerken, die jeweils für sich getrennte Anforderungen besitzen, etwa weil ein Sub-Netz in der Produktion, ein anderes in der Verwaltung und ein weiteres im Vertrieb eingesetzt wird. Sind Produktion und Verwaltung meist lokal organisiert, sind einzelne Vertriebsstellen oft über mehrere Orte verteilt, in großen Konzernen häufig weltweit. Hier bieten sich die Softwaremanagementlösungen an, die sehr flexibel auf verschiedene Infrastrukturen reagieren können.

1.6 Das Windows Update

Das Update Verfahren von Microsoft für die Non-Server Windows Betriebssysteme ist ein wahrer Allrounder. Es installiert sehr bequem Updates für das Betriebssystem, Treiber für Hardware, die Microsoft Office Palette, den Internet Explorer, den SQL Server und andere Software. Außerdem authentifiziert es die Windows Kopie des Nutzers mittels dem *Microsoft-Genuine-Advantage-Programm*.

Windows Update installiert einen Dienst, der ständig im Hintergrund läuft und Kontakt zum Microsoft Patch Server hält, den **Windows Update AutoUpdate Service**, kurz **wuauserv**. Je nach Einstellung der Update Funktionalität kann der Dienst Updates automatisch suchen oder überhaupt nicht. Man kann bestimmen, ob Updates automatisch installiert oder erst heruntergeladen und dann manuell installiert werden oder nur manuell heruntergeladen und installiert werden sollen (Vgl.: Abbildung 14). Ist die automatische Installation gewählt, kommuniziert der Dienst selbstständig mit dem Update-Server und kann selbstständig Updates installieren. Der User bekommt nur ein Pop-up im System Tray mit der Mitteilung, dass Updates installiert werden.

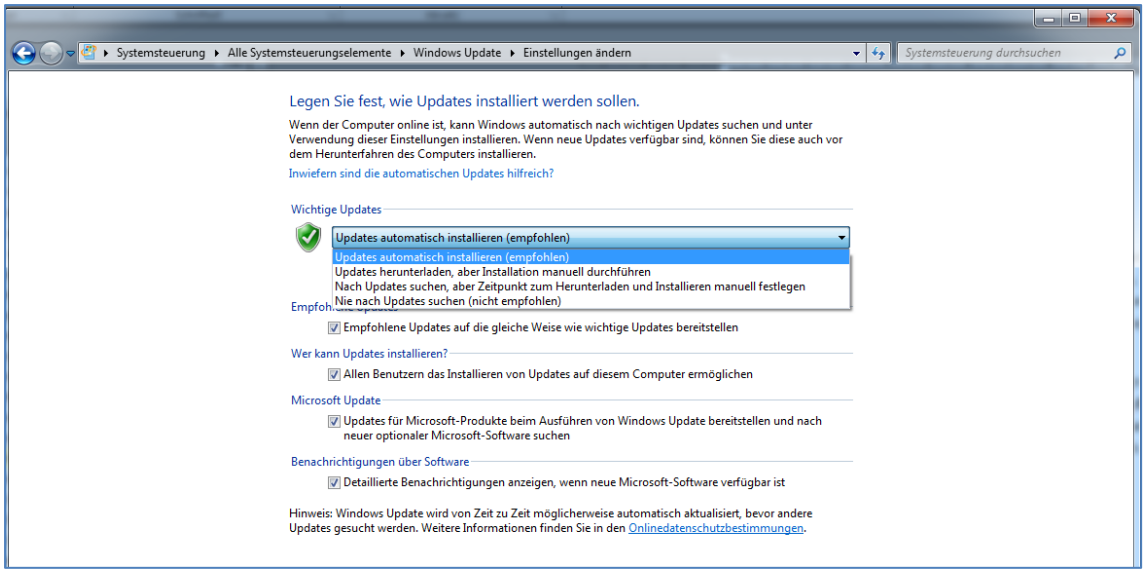

Abbildung 14: Das Windows Update kann auf Wunsch vollautomatisch, ganz oder teilweise manuell ablaufen

1.7 Ökonomische Nutzung

Die Instandhaltung und Aktualisierung von Software weist sich nicht nur durch den technischen Nutzen aus. So spielen auch wirtschaftliche Faktoren eine Rolle. Die Ziele bestehen aus Kundengewinnung, den Erhalt der Kundschaft und den maximal möglichen Verkaufswert der eigenen Produktreihe.

Dem Kunden werden diverse Angebote unterbreitet, um ihm die Attraktivität des Anbieters aufzuzeigen und ihn nach Möglichkeit zum Kauf von zusätzlichen Leistungen zu bewegen. Das hat dann für beide Seiten Vorteile: Der Anbieter erhöht seinen Gewinn, wobei sich der Kunde Vorteile für seine Software erkauft.

1.7.1 Das Upgrade

„**Upgrade** [ˈʌpˌgreɪd] (englisch upgrade = Aufrüstung, Verbesserung) bezeichnet eine Steigerung der Nützlichkeit oder Qualität von z. B. Hard- und Software.“[54]

Das Software Upgrade verbessert die Leistungsfähigkeit einer Software. Die Kennzeichnung des Upgrades ist die Erhöhung der Versionsnummer. Es werden neue Funktionen installiert, die sich von der alten Version spürbar absetzen sollten. Man unterscheidet hier grundsätzlich zwischen *Minor Upgrades* und *Major Upgrades*.

[54] Vgl.: http://www.hardware-aktuell.com/lexikon/Upgrade/

1.7.1.1 Minor Upgrades

Minor Upgrades enthalten kleinere Verbesserungen der Funktionalität von Programmen und die Entfernung von Fehlern. Sie werden meistens kostenlos angeboten. Hierbei ändern sich nur die Nachkommastellen der Versionsnummer. [55]

1.7.1.2 Major Upgrades

Major Upgrades spielen grundlegende Veränderungen für die Software ein. Sie sind in der Regel kostenpflichtig. Sie beinhalten Fehlerbehebungen und neue Funktionen. Außerdem werden Anforderungen an das Programm angepasst. Ein Beispiel dafür ist das Upgrade von *Adobes Photoshop CS 5 (12.0)* auf Version *CS 6 (13.0)*. Hier wurden viele neue Features eingebaut: *Mercury Grafik-Engine, Intuitive Erstellung von Videos, Migrieren und Weitergeben von Vorgaben*, *10-Bit-Farbtiefe* und mehr.[56] Die neue Version ist kostenpflichtig und wird als Upgrade vergünstigt angeboten. Die Vollversion ist für € 799,00, das Upgrade für € 229,00 (Stand: Juli 2012, exkl. MwSt.) erhältlich.[57] Damit beträgt der Preis für das Upgrade etwa 29% des Neukaufpreises. Diese Art der Vermarktung sorgt für eine gute Kundenbindung. Außerdem wird ein Abonnement angeboten, bei dem automatisch immer die aktuelle Version verfügbar ist.

Ebenfalls wird die Aufwertung von der Demoversion einer Applikation zur Vollversion als Major Upgrade bezeichnet. Ein Beispiel kann die Aufwertung einer freien Antiviren Software mit eingeschränkten Features zur kostenpflichtigen Vollversion sein. Hier verdient der Verkäufer der Software, indem er eine eingeschränkte, kostenlose Version anbietet, die dem Kunden die Applikation vorstellt und die er eventuell zeitlich begrenzt testen kann. Genau diese Möglichkeit zum Testen ist der Vorteil, den der Kunde nutzen kann. Beim Kauf des Produkts stehen ihm dann alle Features zur Verfügung.

1.7.2 Das Downgrade

In einigen Fällen kann eine Rückabwicklung eines Upgrades oder die Installation einer Vorgängerversion nötig sein. Das wird dann als *Downgrade*[58] bezeichnet. Die Gründe dafür können vielseitig sein. Der häufigste Fall für die Notwendigkeit eines Downgrades ist die Inkompatibilität der Software mit dem Betriebssystem oder der Hardware des Benutzers.

[55] Vgl.: http://www.hardware-aktuell.com/lexikon/Upgrade/
[56] http://de.wikipedia.org/wiki/Adobe_Photoshop
[57] http://www.adobe.com/de/products/photoshop.html
[58] Vgl. Joos, Planungsbuch Microsoft-Netzwerke, 2006, S.56

Ebenfalls kann eine Unausgereiftheit der Software ein Grund sein. Wenn Fehler auftreten, die nicht zügig gepatcht werden können, muss ein Downgrade zur älteren Version stattfinden.

1.7.3 Das Cross Update

Dem Cross Update[59] liegt eine Marketing Strategie zugrunde, bei der kein Update im eigentlichen Sinn durchgeführt wird. Vielmehr werden potentielle Kunden durch erhebliche Preissenkungen dazu verlockt, Produkte konkurrierender Firmen zurückzugeben und das eigene Produkt zu erwerben. Diese sollten sich natürlich sehr ähnlich sein. So wirbt beispielsweise der Onlineshop „comwave.de“ mit der Navigationssoftware „Destinator 7 - Cross-Update - West- und Osteuropa“[60] als Upgrade für seine Vorgängerversion und bietet sie zusätzlich als Ersatz für vom Kunden genutzte Navigationssoftware von anderen Herstellern an. Dafür muss der Kunde lediglich den Kaufbeleg an „comwave.de“ faxen und erhält dafür einen Rabatt von knapp 30 Prozent.

1.8 Sicherheitsrisiken von Updates

Bei der Verteilung von Updates über das Internet können, wie bei jedem anderen Datentransfer im Netz, Sicherheitsrisiken entstehen. Eine Bedrohung der Sicherheit kann verschiedene Auswirkungen haben. Es kann zu Fehlfunktionen der Software, wie Programmabstürzen oder veränderten Daten, kommen. Informationen wie Passwörter können ausspioniert oder PCs in Botnetze assimiliert werden. Programme, die Informationen im infizierten System sammeln, werden als **Spyware** bezeichnet.

1.8.1 Risiken und deren Entstehung

Sicherheitsrisiken entstehen meist durch den technischen Zustand von Hard- und Software[61], wobei Bedrohungen durch den schlechten Zustand von Hardware meist in Zusammenhang mit dem Alter der Systeme und deren Wartung steht. Im Kontext von Softwareupdates sind für uns die Entstehungen von Sicherheitsrisiken über Probleme mit der Software interessant, weshalb

[59] Vgl. Koch, Computer-Vertragsrecht, 2009, S.441
[60] http://www.comwave.de/GPS---Navigation/Destinator/Destinator-Update-PDA/Destinator-7---Cross-Update---West--und-Osteuropa.html
[61] Vgl.: Studer, Netzwerkmanagement und Netzwerksicherheit, 2010, S. 58ff

Hardwarerisiken hier nur nebenbei erwähnt werden sollen. Risiken durch hardwarebedingte Unzulänglichkeiten sind eher Probleme der Kundenseite bei Updates.

Wie können Sicherheitslücken in Softwareanwendungen entstehen? Man kann davon ausgehen, dass keine Software absolut sicher ist. Alle großen Softwarehersteller wie *Microsoft*, *SAP* oder *Adobe* bringen regelmäßig Sicherheitsupdates für ihre Programme auf den Markt. Diese sind nötig, da Software oft Programmierfehler in Form von Sicherheitslücken enthält, die Angreifern Optionen zum Zufügen von Schadsoftware bringen. Diese Schadsoftware wird im Zusammenhang mit Sicherheitslücken als **Exploit** bezeichnet. Meist zielt Sie darauf hin, Pufferüberläufe auszunutzen. Dabei werden zu große Datenmengen in den Speicherbereich geschrieben, der für das Programm reserviert wurde. Dann kann es vorkommen, dass Speicherblöcke überschrieben werden, die nach dem Ziel-Speicherbereich des Programms liegen, welche zum Beispiel für das Betriebssystem relevant sind und damit kompromittiert werden können.

Außer der Nutzung von Sicherheitslücken innerhalb der Software können auch an Software angehängte oder eigenständig gesendete Dateien Schadsoftware enthalten. Diese wird als **Malware** bezeichnet. Dazu zählen unter anderem Viren, Würmer und Trojanische Pferde, welche auch als Trojaner bekannt sind. Sie sind eigenständig handelnde Programme, die das angegriffene System infizieren können.

Das Opfer solcher Angriffe zu werden, ist für den Nutzer oft nicht leicht zu erkennen, da die Schadsoftware oft im Hintergrund agiert und den Nutzer nicht bei seiner Arbeit behindert oder sonstige Auffälligkeiten nachweist. Das trifft vor allem auf *Spyware* zu. Doch wie erkennt der Nutzer einen Angriff und wie hoch ist die Wahrscheinlichkeit, angegriffen zu werden? Für die Erkennung von Attacken gibt es verschiedene technische Möglichkeiten, die unter anderen im Kapitel 1.8.4 erläutert werden. Für die Wahrscheinlichkeit, überhaupt angegriffen zu werden, gelten laut Prof. Dr. Bruno Studer drei Regeln: „Je leichter der Zugriff auf die Ressource ist, desto größer ist die Eintrittswahrscheinlichkeit eines Angriffes. Je wertvoller die Ressource ist, desto größer ist die Eintrittswahrscheinlichkeit. Je kleiner der Aufwand für einen Angriff ist, desto größer ist die Eintrittswahrscheinlichkeit.“[62] Studer meint damit, dass Angreifer immer versuchen werden, mit dem möglichst geringsten Aufwand den möglichst größten Schaden zu verursachen. Daher sollte bei der Softwareverteilung darauf geachtet werden, es dem Angreifer

[62] Vgl.: Studer, Netzwerkmanagement und Netzwerksicherheit, 2010, S. 60

so schwer wie möglich zu machen und die technischen Möglichkeiten[63] zur Prävention und zur Beseitigung von Schadsoftware zu nutzen.

1.8.2 Mögliche Angriffsarten

Um Sicherheitslücken zu verstehen und Angriffen vorbeugen zu können, muss man sich zunächst damit beschäftigen, wie Angreifer vorgehen und welche Möglichkeiten sie haben, um ihre Ziele zu erreichen.

Sehen wir uns dazu zunächst die Problematik der **Malware** an. Malware ist Software, die im Gegensatz zum Exploit Schaden auch ohne vorhandene Sicherheitslücken ausrichten kann. Sie wird in verschiedene Kategorien unterteilt.

1.8.2.1 Viren und Würmer

Der berühmteste Vertreter der Malware ist der **Virus**. Der Virus ist dadurch charakterisiert, dass er ständig Kopien von sich selbst anfertigt und diese in Programme, Dokumente oder Datenträger schreibt. Damit ähnelt er dem biologischen Virus, der sich als Parasit mithilfe von Wirten fortpflanzt. Viren zielen meist daraufhin, sich in Computern einzunisten, so oft wie möglich zu vervielfältigen und im Dateisystem den größtmöglichen Schaden zu verursachen. Ebenso ist es möglich, dass Schäden an der Hardware oder dem Betriebssystem vorgenommen werden. Oft dienen sie dazu, die Sicherheit eines PCs zu beeinträchtigen.

Der **Wurm** ist im Gegensatz zum Virus nicht von Wirtssoftware abhängig. Er muss nicht passiv warten, bis ein Wirtsprogramm vom User gestartet wird, sondern versucht sich aktiv über Netzwerke oder Wechselmedien wie USB Sticks zu verbreiten. Dazu nutzt er Sicherheitsprobleme im Zielsystem, die oft durch den Nutzer selbst hervorgerufen werden. Er verbreitet sich in Netzwerken vor allem per E-Mail, in denen er als Anhang oder per Internetlink auf das Zielsystem gelangt oder über Instant Messenger wie *ICQ* oder *MSN Messenger*, bei denen User einen Link erhalten, der bei Klick auf den Ort des Wurms im Internet verweist, damit dieser heruntergeladen werden kann. Auf die gleiche Art werden Würmer über das *Internet Relay Chat (IRC)* oder über Netzwerkressourcen verteilt.

1.8.2.2 Trojanische Pferde

Trojanische Pferde, auch als Trojaner bekannt, sind Schadsoftware, die sich in scheinbar nützlichen Programmen tarnen. Meistens werden Sie im Internet durch Cracks oder

[63] Vgl.: Kapitel 1.9.4 Technische Möglichkeiten

Keygeneratoren verbreitet. Sie unterscheiden sich durch die Aktionen, die sie auf dem infizierten PC ausführen sollen. Die wichtigsten werden in den folgenden Abschnitten erläutert.

Backdoor Trojaner sind Dienstprogramme zur Fernverwaltung von Computern in einem Netzwerk. Im Gegensatz zu kommerziellen Fernverwaltungstools wie TeamViewer[64] oder PCVisit[65] weiß der Nutzer nichts vom Vorhandensein der Software. Sie ist meistens nicht einmal in der Prozessliste eingetragen. Der Trojaner kann auf sämtliche Systemvorgänge des infiltrierten Systems zugreifen. Somit kann der Angreifer den PC per Fernsteuerung benutzen, um zum Beispiel Daten zu empfangen oder zu senden, zu öffnen oder zu löschen. Damit kann er auch persönliche Daten des Opfers ausspionieren und andere Viren installieren oder auch Funktionen wie das Herunterfahren des PCs ausführen. Backdoor Trojaner können ähnlich wie Würmer im Netz verbreitet werden, jedoch nicht selbstständig, sondern auf Befehl des Angreifers hin, der den Trojaner steuert. [66]

Password Stealing Ware (PSW) sind Trojaner, die zum Diebstahl von Informationen auf dem Zielcomputer benutzt werden. Das sind meistens Passwörter des Nutzers, die sich in persönlichen Dateien befinden. Auch können Daten wie Telefonnummern oder E-Mail-Adressen gestohlen werden. Selten spionieren PSW Trojaner auch Systeminformationen aus. Das können Daten über das verwendete Betriebssystem, Hardwaremerkmale oder installierte Software mit deren Registryeinträgen oder Zugangscodes zu Onlinespielen sein.[67]

DDoS Trojaner sind durch ihre starke Vermehrung im Internet gekennzeichnet. Sie sind nicht darauf konzipiert, Schaden am infizierten PC anzurichten, sondern von diesem aus bei Aktivierung durch den Angreifer eine sogenannte **Distributed Denial of Service** (DDoS) Attacke auszuführen. Dabei werden zahlreiche Anfragen an einen Webserver gesendet. Geschieht dies gleichzeitig von möglichst vielen Computern aus, kann es zum Zusammenbruch des Servers führen. In der Regel wird diese Methode verwendet, um gezielt Homepages auszuschalten. Dadurch, dass die Angriffe von willkürlichen PCs ausgehen, die weltweit verteilt sind, kann der Angreifer schwer identifiziert werden. Da die Nutzer der PCs, von denen die Angriffe gestartet werden, sich selbst im Unklaren darüber sind, durch einen Trojaner zu der Attacke benutzt zu

[64] Vgl.: http:// www.teamviewer.com
[65] Vgl.: http://www.pcvisit.de/
[66] Vgl.: Kasperskij, Malware, 2008, S. 63f.
[67] Vgl.: Kasperskij, Malware, 2008, S. 64

werden, werden Sie in ihrer Gesamtheit als *Botnetze* bezeichnet, die ohne es zu wissen, ferngesteuert Angriffe auf Server starten.[68]

Trojan Spies überwachen den Nutzer des infizierten Computers. Sie registrieren Tastatureingaben, fertigen Screenshots an und senden diese Daten an den Angreifer. Meist werden sie dazu benutzt, Daten vom Onlinebanking des Opfers oder Zugangsdaten von anderen Zahlungssystemen im Internet auszuspionieren.[69]

1.8.2.3 Netzwerk-Installer und Updatesicherheit

Netzwerk-Installer sind Programme, die zum Download und zur Installation von Software über Netzwerke vorgesehen sind.[70] Vertreter der Netzwerk-Installer ist zum Beispiel der *Softonic Download Manager*[71]. Sie dienen dafür als Container und sind durch das Ausführen von Dateien sicherheitstechnisch bedenklich. Es ist für Angreifer nicht nötig, selbst Programme auf PCs einzuspielen denn die Netzwerk-Installer können als legale Tools zum Ausführen von Malware dienen. Dazu könnte die Netzwerkressource, auf die der Installer zugreift, verändert oder die Adresse geändert werden. Dann würde der Installer, ohne dass er und der Nutzer es merken, Malware installieren.

Unter die gleiche Kategorie der Gefährdung zählen Updatetools. So soll der Wurm *Worm.Win32.Flame,* der als Flame bekannt ist, unter anderem über das Windows Update verteilt worden sein. Microsoft hatte bestätigt[72], dass Flame mit gültigen Microsoft Zertifikaten signiert war und somit offiziell als unbedenklich vom Betriebssystem eingestuft wurde. Damit wäre es theoretisch auch möglich gewesen, den Wurm über das Windows Update zu erhalten.

Für Unternehmen, die Software über Netzwerke, speziell das Internet, verteilen, gilt es daher besondere Vorsichtsmaßnahmen zu treffen. Darüber hinaus sind einige Gesetze und Sicherheitsvorschriften zu beachten. Einige davon werden im nächsten Kapitel erläutert.

[68] Vgl.: Kasperskij, Malware, 2008, S. 65
[69] Vgl.: Kasperskij, Malware, 2008, S. 68
[70] Vgl.: Kasperskij, Malware, 2008, S. 71
[71] Vgl.: http://www.softonic.de/download-manager
[72] Vgl.: http://www.heise.de/security/meldung/Super-Spion-Flame-trug-Microsoft-Signatur-1590335.html

1.8.3 Sicherheitsregeln und Standards

Für die IT-Sicherheit gelten verschiedene Gesetze, Verordnungen, Vorschriften und Anforderungen. Im Rahmen dieser Diplomarbeit möchte ich mich auf die deutschen Vorschriften und Gesetze für Unternehmen beschränken.

In Deutschland ergeben sich für Unternehmen in Hinblick auf *Sorgfalt und Haftung der Geschäftsleitung* gesetzliche Anforderungen. Beispiele dafür sind das GmbH-Gesetz, §43, Haftung der Geschäftsführer oder das Aktiengesetz (AktG), §93, Sorgfaltspflicht und Verantwortlichkeit der Vorstandsmitglieder. Das Bürgerliche Gesetzbuch (BGB) geht in § 280 auf Schadensersatz wegen Pflichtverletzung ein.[73]

Dazu wurden Standards und Rahmenwerke entwickelt, die Institutionen und Unternehmen unterstützen, indem sie helfen, „dass allgemein anerkannte, einheitliche Methoden und Best Practices in die Realisierung von Informationssicherheit einfließen“[74]. Die Umsetzung dieser Standards ist in der Regel zertifizierbar, sie lassen sich auch kombinieren.

Ein wichtiger Standard ist der **IT-Grundschutz**. Er wurde vom Bundesamt für Sicherheit für Informationstechnik (BSI) entwickelt und stellt Methoden zur Verfügung, um „individuelle Informationsverbünde abzugrenzen und für […] deren Bestandteile angemessene Sicherheitsmaßnahmen auswählen [zu] können“[75]. Diese Informationsverbünde sind zum Beispiel abgetrennte Strukturen eines Unternehmens, wie das Personal, die Technik oder die Organisation. Diese Bereiche sollten möglichst getrennt voneinander betrachtet werden und haben individuelle Sicherheitsbedürfnisse. Dabei werden Maßnahmen als kontinuierliche Anpassungen und Verbesserungen des Sicherheitsmanagements gesehen. Der IT-Grundschutz enthält verschiedene Kataloge, die Implementierungshilfen für Sicherheitsprozesse beinhalten. Dazu zählen unter anderem Bausteine wie Infrastruktur, Netze oder Anwendungen, Gefährdungskataloge zur Risikoeinschätzung, Maßnahmenkataloge und Kataloge zur Erstellung und Umsetzung von Sicherheitskonzepten.

Neben dem IT-Grundschutz existieren diverse Normen für das IT Management, wie die ISO 2000 oder ITIL, die zum Erhalt von Zertifikaten helfen, indem sie Anforderungen zur kontinuierlichen Verbesserung von Prozessen vorgibt.

[73] Vgl.: Müller, IT-Sicherheit mit System, 2011, S. 27
[74] Vgl.: BSI, Informationssicherheit, 2009, S.7
[75] Vgl.: BSI, Informationssicherheit, 2009, S.10

1.8.4 Technische Möglichkeiten

1.8.4.1 Schutz des Nutzers

Zum Schutz vor Malware und anderen Angriffen kommen verschiedene Möglichkeiten zum Einsatz. Dazu zählen auf Nutzerseite zuverlässige Antivirensoftware und eine Firewall, um sich gegen Netzangriffe zu schützen. Außerdem sind integrierte Sicherheitslösungen erhältlich, die zusätzlichen Schutz gegen Angriffe aus dem Netz, Filter gegen Spam oder Schutz gegen den Besuch unerwünschter oder gefährlicher Internetseiten bieten. Wichtig ist hierbei, dass die Tools zur Systemsicherheit immer auf dem aktuellen Stand sind, da die Hersteller auf aktuelle Gefahren reagieren müssen und daher ständig Patches verteilen.

Ebenfalls ist zur Vorsicht bei Internetlinks, Emails oder Nachrichten von Instant Messagern geraten. Hier kann es sich um Trojaner handeln oder um einen Wurm, der sich von einer falschen Emailadresse aus verbreitet.

1.8.4.2 Schutz des Herstellers und des Updates

Mit den vorgenannten Maßnahmen können Nutzer sich selbst vor Angriffen schützen. Doch gerade bei Updates ist das schwierig, da die Updatetools in der Regel Firewallfreigaben benötigen und somit im Falle von Malwareinfektionen ungehinderten Zugang zum Client PC haben. Dann kann nur noch die Antivirensoftware auffällige Programme finden und eliminieren. Das bedeutet im Umkehrschluss, dass der Verteiler der Updates gehalten ist, seine Programme sicher und von Malware frei zu halten. Dazu muss die Software in erster Linie vor Exploits gesichert werden. Das kann mit *Intrusion Detection* Systemen und *Intrusion Prevention* Systemen realisiert werden, die im Zusammenspiel Angriffe erkennen und verhindern können. Die Intrusion Detection Systeme erkennen über verschiedene Algorithmen, ob ein Angriff stattfindet. Dabei werden Angriffssignaturen und Filter verwendet, um spezifische Angriffsmuster zu erkennen. Die Intrusion Prevention Systeme können darüber hinaus noch dafür sorgen, dass ein Angriff blockiert wird. Sie können Datenströme unterbrechen und Regeln von Firewalls verändern.

Das Hauptproblem ist jedoch die Entstehung von Exploits. Sie findet bereits in der Entwicklungszeit der Anwendung statt. Denn Exploits entstehen durch Fehler in der Software, durch unsaubere Programmierung oder generelle Fehler. Sie zu verhindern ist die Aufgabe des Softwareentwicklers, der gerade Fehler im Zusammenhang mit Pufferüberläufen vermeiden sollte.

Neben der Entstehung von Exploits muss auf die Sicherheit der zu transportierenden Daten geachtet werden. Dazu muss der Server, auf dem sich die zum Update bereitstehenden Dateien befinden, gut vor Angriffen geschützt werden. Das kann durch Firewalls oder durch die Verschlüsselung der Dateien geschehen. Ebenso kann der Zugriff auf den Server eingeschränkt werden, so dass nur Entwicklern Schreibzugriff auf die Dateien ermöglicht wird. Zusätzlich kann der Server noch getrennt vom Firmennetzwerk eingerichtet werden, damit er von Angriffen auf dieses verschont bleibt. Im Umkehrschluss bleibt auf diese Weise das Firmennetzwerk von Angriffen auf den Server geschützt.

1.8.4.3 Protokolle und Tools

Um die Daten vom Hersteller auf den Server für die Updates zu transportieren, können sichere Verbindungen mit *SSL/TLS* oder *Tunnel* benutzt werden. **Transport Layer Security** (TLS) ist als Nachfolgerprotoll von SSL (Secure Sockets Layer) zur Verschlüsselung von Datenübertragungen innerhalb der Transport Schicht im OSI Modell[76] bekannt. TLS ermöglicht unter anderem Server- und Clientauthentifizierung, Verschlüsselung und Integrität der Daten.[77]

Daneben existieren noch Protokolle zur Sicherheit für andere OSI-Schichten. Dazu gehört das Protokoll **S-http** der Anwendungsschicht, das ebenfalls verschlüsselte Übertragungen ermöglicht.

Auf der Internet Schicht arbeitet **IPsec** (Internet Protocol Security), ein Tool, das einzelne IP-Pakete verschlüsseln und Header zur Identifikation anfügen kann. Es kann ebenfalls zum Aufbau virtueller Netzwerke (VPN)[78] verwendet werden.

Das Extensible Authentication Protocol (EAP) dient zur Authentifizierung, das vor allem im 802.1X Standard, also im WLAN Bereich zur Anwendung kommt. Dabei können zur Authentifizierung Kombinationen aus Benutzernamen und Passwort, Einmal-Passwörtern oder Zertifikaten genutzt werden.[79] Es wird aber auch bei der Fernwartung oder zur Tunnelung verwendet.

1.8.4.4 Virtuelle Netzwerke

Ein **Virtual Private Network** (VPN) virtualisiert, wie der Name vermuten lässt, ein privates Netzwerk. Private Netzwerke sind Netzwerke, die den Zugang nur limitierten Personen bzw. Computern gewähren. Was für LAN (Local Area Network) sehr leicht ist, ist im Internet schwer

[76] Vgl.: Kapitel 2.4.2.1
[77] Vgl.: Exner, Secure Socket Layer (SSL), 2008, S. 3
[78] Vgl.: Kapitel 1.8.4.4 Virtuelle Netzwerke
[79] Vgl.: Schwenkler, Sicheres Netzwerkmanagement, 2005, S. 220

realisierbar, da Standleitungen sehr teuer sind. VPNs können ein privates Netzwerk über das Internet aufbauen. Dafür definieren sie einen Tunnel zwischen dem Router einer Seite und dem Router der anderen Seite und nutzen IP-in-IP Verkapselung, um über den Tunnel Datagramme zu übertragen.[80] Dazu werden Protokolle wie *IPsec* oder *TLS* verwendet.

Durch die IP-in-IP Technik sind VPNs sehr sicher. Dazu wird das innere Datagramm inklusive Header verschlüsselt, bevor es gekapselt wird. (s. Abbildung 15). Beim Empfang der Daten wird dann der Datenbereich entschlüsselt und das innere Datagramm wiederhergestellt. Der dafür nötige Schlüssel ist nur im VPN selbst bekannt, selbst die Datenquelle und das Ziel sind verborgen, da der Header des inneren Datagramms ebenfalls verschlüsselt wird.

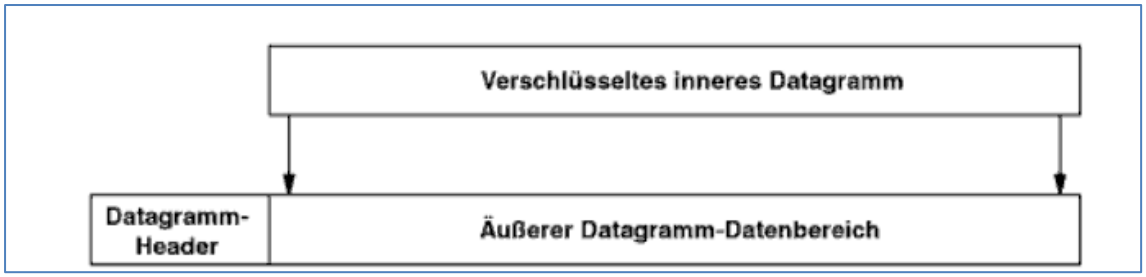

Abbildung 15: Ip-in-IP Kapselung virueller Netzwerke[81]

1.8.4.5 Digitale Signaturen und Zertifikate

Digitale Signaturen weisen den Sender von Daten gegenüber dem Empfänger aus. Die Vorgehensweise dazu ähnelt einer kryptischen Form der Prüfsumme. Um eine solche Signatur zu generieren, benutzt der Absender einer Nachricht einen privaten Schlüssel. Um diesen überprüfen zu können, erhält der Empfänger einen öffentlichen Schlüssel. Mit diesem öffentlichen Schlüssel kann jeder den privaten überprüfen und daran den Absender erkennen. Dazu muss allerdings das signierte Dokument unverändert bleiben, andernfalls schlägt die Überprüfung fehl.[82]

[80] Vgl.: Corner, Konzepte, Protokolle, Architekturen, 2011, S. 376
[81] Vgl.: Corner, TCP/IP, 2011, Abb. 20.2
[82] Vgl.: Schmeh, Kryptografie, 2009, S. 31

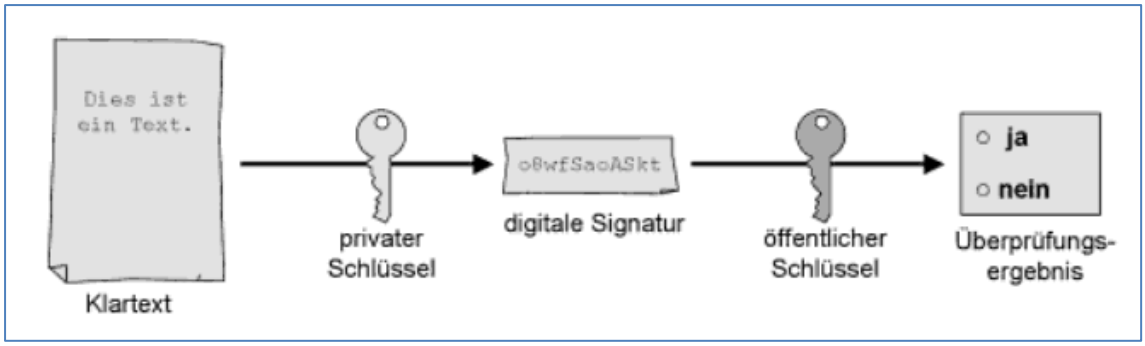

Abbildung 16: Anhand digitaler Signaturen können Nutzer erkennen, ob ein Dokument unverändert von einem sicheren Absender gesendet wurde.[83]

Der Vorteil dieser asymmetrischen Verschlüsselungstechnologie gegenüber symmetrischen mit geheimen Schlüsseln ist, dass der öffentliche Schlüssel bedenkenlos über unsichere Kanäle gesendet werden kann. Jedoch ist hier zu beachten, dass dem öffentlichen Schlüssel nicht anzusehen ist, wie lange er gültig ist und von wem er herausgegeben wurde. So könnte ein Angreifer seinen eigenen öffentlichen Schlüssel als den eines anderen ausgeben.

Um den Herausgeber eines öffentlichen Schlüssels eindeutig identifizieren zu können, wurde das Verfahren der **digitalen Zertifizierung** eingeführt. Digitale Zertifikate sind Datensätze, die Informationen zu öffentlichen Schlüsseln enthalten. Dazu gehört der Name des Herausgebers, den Gültigkeitszeitraum des Schlüssels sowie Verwendungszweck und andere Informationen. Damit kann eindeutig der Herausgeber des Schlüssels sowie seine Gültigkeit festgestellt werden.

Um Zertifikaten ihrerseits Gültigkeit zu geben, werden diese ebenfalls digital signiert. Diese Signaturen werden von öffentlichen *Zertifizierungsstellen (engl.: Certification Authority[CA])*, also digitalen Ausweisstellen, herausgegeben. Um die Zertifikate nutzen zu können, werden noch weitere Komponenten benötigt wie zum Beispiel ein Server für den Aufruf der Zertifikate und eine Registrierungsstelle.[84] Die Infrastruktur, die aus diesen Komponenten besteht und die digitalen Zertifikate verbreitet, wird als *Public-Key-Infrastruktur* (PKI) bezeichnet. Es ist sinnvoll, das Zertifizierungsstellen ebenfalls ein digitales Zertifikat besitzen, welches von anderen Zertifizierungsstellen herausgegeben wird. So entsteht eine Art Hierarchie, deren Kopf eine Wurzel-Zertifizierungsstelle, in der Fachwelt *Root-CA* genannt, ist (siehe Abbildung 17).

[83] Vgl.: Schmeh, Kryptografie, 2009, Abbildung 3.2
[84] Vgl.: Schmeh, Kryptografie, 2009, S. 34

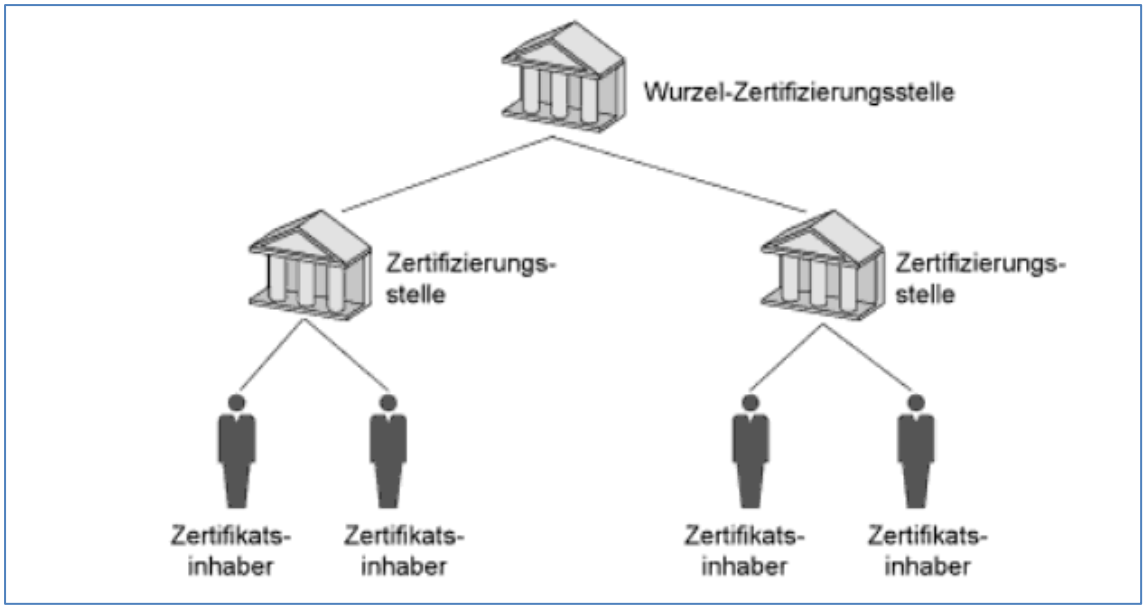

Abbildung 17: Hierarchie von Zertifizierungsstellen[85]

[85] Vgl.: Schmeh, Kryptografie, 2009, Abbildung 3.3

1.8.5 Fazit

Welche technischen Möglichkeiten zu welchem Zweck sinnvoll sind, ist immer die Entscheidung des Verteilers von Updates oder anderer Daten. Hier spielen verschiedene Faktoren eine Rolle. Die Nutzung von Zertifikaten ist eine finanzielle Frage, die Virtualisierung von Netzwerken hingegen eine technische. Welche Tools und Techniken von einem Softwarehersteller genutzt werden sollten, hängt immer von einer Risikobetrachtung ab. Dazu können die drei Regeln der Eintrittswahrscheinlichkeit von Angriffen[86] und ein möglicher wirtschaftlicher Schaden betrachtet werden. Auch ist eine Kalkulation der Beeinträchtigung des Images des Unternehmens durch Hackerattacken in die Risikobetrachtung mit einzubeziehen.

Sinnvoll ist meistens, zunächst den Schutz des Nutzers und den des Updateservers zu gewährleisten. Wenn die auf dem Server befindlichen Dateien gut geschützt und die PCs der Nutzer sicher sind, können viele Angriffe vermieden werden. Dazu ist für den Kunden wichtig, dass er regelmäßig Sicherheitsupdates seiner Software und des Betriebssystems installiert. Die meisten Hackerangriffe können dadurch präventiv abgewehrt werden. Im Zusammenhang mit den Updates von Softwareherstellern und Exploits ist es für den Kunden ebenfalls wichtig, diese Programme immer aktuell zu halten. So können erkannte Schwachstellen durch gut gepflegte Software geschlossen und die Anwendungen sicher erhalten werden.

Ob die Einrichtung von VPNs oder die Nutzung von Signaturen, mit oder ohne Zertifikat, sinnvoll ist, muss von Fall zu Fall betrachtet werden. Große Firmen, die viele Daten über das Internet verteilen, profitieren sicherlich von diesen Möglichkeiten.

[86] Vgl.: Kapitel 1.8.1

2 Konzeption eines Update Systems

2.1 Tätigkeitsbereich

Im Rahmen dieser Diplomarbeit soll in Zusammenarbeit mit dem Unternehmen **Vordruckverlag Weise GmbH**[87] ein Programm entwickelt werden, das Updates für die Software des Unternehmens für Kunden bereitstellen soll.

Der vom Geschäftsführer Ingo Weise geleitete Vordruckverlag ist eine in Dresden ansässige Firma, die 1993 gegründet wurde, um für Architekten und Bauingenieure Vordrucke und Formulare bereit zu stellen. In den folgenden Jahren wurde das Angebot kontinuierlich erweitert, bis 1996 der Entschluss gefasst wurde, eine Software zur Formularerstellung zu entwickeln.

Danach entstanden eine Datenbankapplikation, die Erweiterung der Formularsoftware und der Aufbau der elektronischen Gesetzessammlung „Baurecht aktuell“.

Durch den Aufbau einer Softwareentwicklungsabteilung ab dem Jahr 2000 konnten für Kunden flexiblere Lösungen gefunden und neue Softwareprodukte an den Markt gebracht werden. Dazu gehören u.a. eine Software zur Bauablaufplanung und Ressourcenüberwachung, eine Software zur Honorarberechnung und -verwaltung, eine Software zum Sicherheits- und Gesundheitsschutz gemäß Baustellenverordnung, eine Software zur Erstellung von Brandschutzkonzepten für alle Bundesländer, eine Software zur Erstellung von Bautagebüchern, eine Software zum Unternehmenscontrolling und eine Software zur Erstellung von Flucht- und Rettungswegeplänen.

Das Unternehmen bietet für viele der Anwendungen Netzwerklizenzen und Servicepakete an. Ebenfalls bestehen neben dem regulären Kundensupport Möglichkeiten zu Beratungen und Onlineschulungen zu den individuellen Produkten.

Gegenwärtig beschäftigt die Vordruckverlag Weise GmbH 15 Mitarbeiter.

Die Software des Unternehmens wird vor allem mit der Programmiersprache Delphi in der Entwicklungsumgebung RAD Studio entwickelt.[88]

[87] Vgl.: http://www.vordruckverlag.de/
[88] Vgl.: Kapitel 3.1 Entwicklungsumgebung

2.2 Aktueller Stand der Softwareverteilung und –aktualisierung

2.2.1 Ist-Zustand

Die Software der VVW GmbH wird per DVD verschickt und zum Download von einem Webserver angeboten. Für die Paketierung wird InstallShield[89] verwendet, welches neben einer *Setup.exe* ein Windows Installer Paket (msi) zur Installation bereitstellt. Damit benötigt die Software Windows ab Version 2000 und kann nicht auf Linux Distributionen oder Mac OS installiert werden. Genutzt wird hier die Variante der Basic MSI Projects[90], da weitere Skripte nicht nötig sind. Bei Client-Server Versionen wird die Software vom Admin auf jedem Client PC separat installiert.

Zur Aktualisierung der Applikationen wird das Programm **XPressUpdate**[91] der Firma **Pixelplanet**[92] verwendet. Dabei wird bei Programmstart die *XPressUpdate.exe* Datei mit Parameter der Programm-ID aufgerufen, woraufhin sie nach Updates für das Programm sucht. Alternativ kann die Exe über einen Menü Aufruf gestartet werden. Gesteuert wird das Tool über eine Backend-Anwendung auf dem Webserver der Firma.

2.2.2 Mängel / Nachteile

Die Mängel liegen nicht am XpressUpdate selbst, sondern daran, dass das Tool nicht optimal an die Bedürfnisse des Unternehmens angepasst werden kann. Es ist ein allgemeines Updatetool für verschiedene Software und daher nur mäßig individualisierbar. Die Anforderungen für das neue Tool werden in den Kapiteln 2.3.1 und 2.3.2 behandelt.

Ungünstig ist im alten Tool, dass Udpates nicht automatisch am Server abgefragt werden können. Eine Silent Installation ist nicht möglich. So muss der Kunde bei jedem Programmstart zunächst Softwareintern gefragt werden, ob er nach Udpates suchen will, da möglicherweise welche vorhanden sind. Bestätigt er, wird das Tool gestartet und sucht seinerseits nach Updates. Selbst wenn der Kunde sein Programm nur neustartet, erhält er jedes Mal wieder die Meldung, dass eventuell Updates bereitstehen. Alternativ könnte man das Update nach Programmstart deaktivieren und den Kunden nur selbstinitiativ über einen Menüaufruf suchen lassen. Jedoch fehlt dann sämtliche Kontrolle, ob Updates überhaupt gesucht werden. Die Kontrolle fehlt

[89] Vgl.: Kapitel 1.3.5
[90] Vgl.: Kapitel 1.3.5
[91] Vgl.: http://www.xpressupdate.de/
[92] Vgl.: http://www.pixelplanet.de/

allerdings auch, wenn der Kunde bei der Startabfrage immer auf „Nein“ klickt. Doch wird er so zumindest darauf hingewiesen.

Der Fakt, dass nicht automatisch nach Updates gesucht wird, ist selbstverständlich nur ein Problem der Handhabung. Es ist einfach umständlich, bei jedem Programmstart eine solche Abfrage zu erhalten und wegklicken zu müssen. Noch umständlicher ist die Programmwahl: Es gibt keine. Das Update Tool erkennt nur das Programm, aus dem es gestartet wurde und sucht nur dafür Updates.

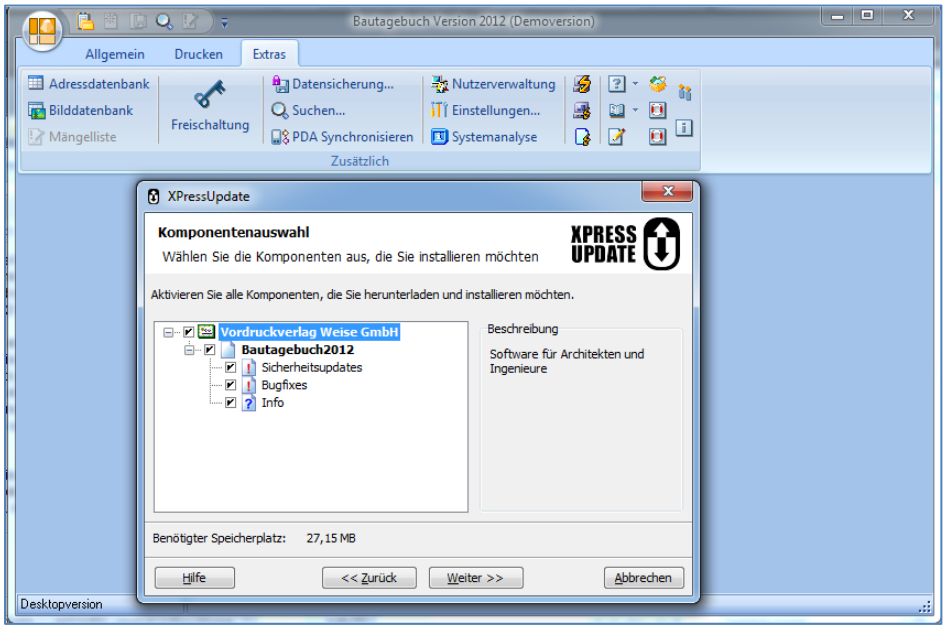

Abbildung 18: Das XPressUpdate kennt nur die Applikation, aus der es gestartet wurde.

Das ist bei der Nutzung mehrerer Programme des Herstellers sehr umständlich. Hier muss das Tool aus jedem Programm einzeln aufgerufen werden und kann nur für genau dieses Programm Updates installieren. Daraus folgt, dass es auch nicht selbständig lauffähig ist und in jeden Programmordner integriert werden muss. Zugute kommt hier dem Tool, dass es sehr klein ist und somit auch in mehreren Programmordnern wenig Platz beansprucht.

Ein weiteres Manko ist die Übertragung des Updates. Möchte ein Entwickler neue Dateien zum Update bereitstellen, muss er diese erst gepackt über eine DLL auf den Webserver des Unternehmens laden. Die DLL unterscheidet, ob es sich bei einer Anfrage um http Request an die Homepage oder die Updateanfrage handelt. Dann wird das Update über Port 80 durchgeführt. Hier stellt sich die Frage der Sicherheit, da auf dem Weg zwischen Entwickler und Webserver Angriffe auf die Dateiübertragung stattfinden können. Der Webserver an sich ist vor Angriffen mit einer Firewall geschützt, kommt die Update Datei aber schon infiziert auf dem

Server an, wird das die Firewall nicht erkennen. Im schlimmsten Fall können dann veränderte Updates an die User weitergegeben werden.

2.3 Zielanforderungen an das Programm

2.3.1 Kundenanforderungen

Das Update Tool soll sowohl für den Softwarehersteller als auch für den Kunden eine Erleichterung gegenüber dem vorhandenen Tool sein. Für den Kunden ist es wichtig, dass er unkompliziert nach Updates suchen und diese automatisch installieren kann. Servicepaketkunden möchten Updates direkt über das Programm erhalten, anstatt den Umweg über die Webseite gehen zu müssen. Administratoren von Client-Server Systemen möchten nicht mehr an jeden Client PC gehen müssen, um dort per Hand zu aktualisieren. Das ist bis jetzt oft erforderlich, da in Delphi Kompilate als fertige .exe Datei verteilt werden. Diese Praxis hat Vor- und Nachteile, die hier allerdings nicht zur Diskussion stehen. Wichtig ist bei diesem Punkt, dass bei fast jedem Softwareupdate die Ausführungsdatei der jeweiligen Applikation überschrieben werden muss, was die ab Windows Vista enthaltene Benutzerkontensteuerung (UAC) des Betriebssystems auf den Plan ruft, die, je nach Einstellung, nach Administratorrechten fragt, um die jeweilige Datei zu überschreiben. Das Gleiche trifft auf Änderungen von Programmdateien zu, die sich im ProgramData Verzeichnis der Windows Partition befinden. Hier werden ebenfalls oft Administratorrechte verlangt, um Zugriff auf Dateien zu erhalten. Von daher muss der Administrator, der seinen Clients nicht automatisch Administratorrechte verleiht, selbst die Installation von Updates vornehmen oder eine andere Lösung finden.

Ein weiterer wichtiger Punkt ist die Erkennung von mehreren Programmen desselben Herstellers. Konnte man bisher nur Updates für das Programm installieren, aus dem das Tool aufgerufen wurde, sollte das neue Programm erkennen, ob noch Updates für weitere Anwendungen bereitstehen und diese nach Wunsch gleich mitinstallieren. Außerdem wäre eine Silent-Installation nach Windows Vorbild[93] und ein Updateplan wünschenswert. Mit diesem könnten dann Zeiträume für automatische Installationen bestimmt werden.

[93] Vgl.: Kapitel 1.3.2, Unattended Installation

2.3.2 Herstelleranforderungen

Den Anforderungen der Kunden gegenüber stehen die des Herstellers. Er möchte eine optimale Performance mit guter Funktionalität und dabei einfacher Handhabung erreichen. Dazu soll das Tool per Nutzererkennung Kundendaten sammeln, mit der internen Datenbank vergleichen und korrekt Updates anbieten. Das bedeutet, dass Updates für Anwendungen jederzeit für die Kunden erreichbar sein sollen, also dass auch Nutzer der Demoversionen Zugang zu Programmupdates erhalten. Servicepackupdates sollen hingegen nur für erworbene Versionen mit zusätzlicher Servicepacklizenz veröffentlicht werden. Außerdem soll ein Offline-Update möglich sein. Dabei kann dann der Admin eine Archivdatei herunterladen und diese auf den Client-PCs installieren, ohne dass diese über eine Internetverbindung verfügen müssen.

2.4 Herangehensweise und Vorbetrachtung

2.4.1 Herangehensweise

Nachdem die Anforderungen für das Projekt *Updatetool* aufgestellt wurden, konnte mit der Entwicklung begonnen werden. Hierfür wurden 3 Projekte in einer Projektgruppe zusammengefasst: Das *UpdateServer* Projekt, das *UpdateClient* Projekt und das *ClientService* Projekt. Widmen wir uns zunächst dem *UpdateServer* und dem *UpdateClient*.

Das *UpdateServer* Projekt erzeugt eine Exe-Datei, die möglichst ohne Unterbrechung auf dem Webserver der Firma laufen soll. Sie lauscht als Serveranwendung auf einem festgelegten Port und reagiert auf Clientanfragen. Dann kann sie über eine Verbindung zur Kundendatenbank des Unternehmens Informationen über die Clients sammeln und entsprechend auf weitere Anfragen reagieren. Auf eine solche hin kann dann verglichen werden, ob die Clients über die aktuelle Versionen der installierten Software verfügen. Wenn nicht, wird aus dem Ordner der jeweiligen Anwendung eine Dateiliste erstellt und dem User zum Download angeboten. Das kann auch für mehrere Anwendungen geschehen.

Das *UpdateClient* Projekt soll beim Start von Anwendungen des Vordruckverlags oder nach Wunsch per Menüaufruf gestartet werden. Es wird eine Verbindung zum UpdateServer hergestellt und nach neuen Updates für die installierten Programme gefragt. Sind welche vorhanden, kann der User auswählen, für welche Anwendungen er Updates installieren möchte und diese dann herunterladen.

Außerdem soll ein Zeitplaner für die Installation von Updates bereitgestellt werden. Damit kann sich der User dann täglich oder wöchentlich nach Neuerungen erkundigen.

Zur Realisierung des Projekts müssen Vorüberlegungen getroffen werden.

2.4.2 Vorbetrachtung

2.4.2.1 Verbindungen mittels Transmission Control Protokoll

Für den Datentransport wird eine Layer 4 Verbindung nach dem OSI – Modell verwendet. Dabei wird das **Transmission Control Protocol** (TCP) verwendet.[94] Das Protokoll sorgt für eine zuverlässige Verbindung zwischen Server und Client.

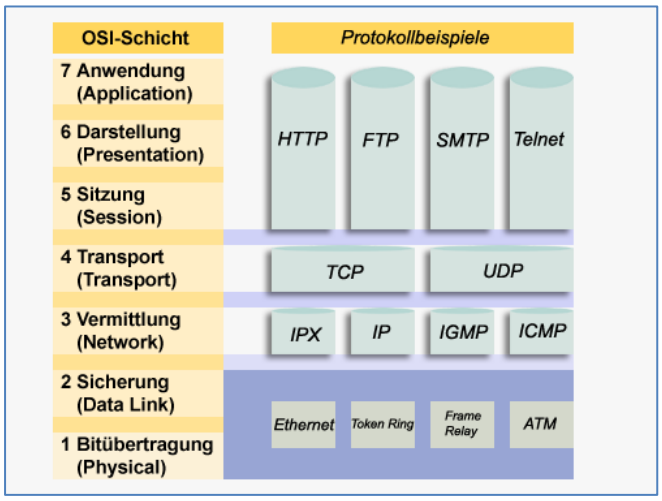

Abbildung 19: Für den Datentransport wird eine TCP Verbindung nach dem OSI Modell verwendet.

Das Transmission Control Protokoll lässt sich durch fünf Eigenschaften charakterisieren:[95]

Ein wichtiger Punkt ist die **Stream-Orientierung**. Streams sorgen dafür, dass die Daten gerade bei größeren Datenvolumina als Bitstrom zwischen zwei User-Prozessen versendet werden. Dabei werden sie in 8 Bit Blöcke, also Bytes aufgeteilt.

Für den Datenaustausch wird von beiden Seiten aus ein Transfer angekündigt, der von Softwareprotokollmodulen (Schicht 3) der beiden Betriebssysteme überwacht wird. Sie sorgen dafür, dass die Daten richtig empfangen werden und können Kommunikationsfehler erkennen und an das Betriebssystem melden. Diese Art der Verbindung wird als **virtueller Schaltkreis** bezeichnet.

[94] Vgl.: Stein, Taschenbuch Rechnernetze und Internet, 2008, Seite 361
[95] Vgl.: Corner, Konzepte, Protokolle, Architekturen, 2011, Seite 216f

Da Datenströme von der Empfängeranwendung in Segmente passender Größe (bis zur Mindestgröße von 8 Bits) aufgeteilt werden und dann sofort verwendet werden können, kann es zu Problemen der Effizienz kommen. So ist es sinnvoller, zunächst Daten zu sammeln und sie dann zu verwenden. Dieser Vorgang wird als **Pufferung** bzw. Buffering bezeichnet. Dabei werden in der Regel genügend Daten gesammelt, um damit ein großes Datagramm zu füllen. Bei großen Paketen kann es aber auch dazu kommen, dass beim Buffering ein Paket in mehrere Segmente aufgeteilt wird. Dazu führt die Überlegung, dass bei fehlerhaften Übertragungen nur ein kleinerer Teil erneut übertragen werden muss und nicht das große Paket.

Strukturierte Daten kann der TCP/IP – Stream nicht bearbeiten wie etwa das *Session Initiation Protcoll*. So kann zum Beispiel keine Trennung zwischen mehreren Datensätzen innerhalb eines Streams erfolgen. Diese Arbeit müssen die Anwendungen des Protokolls vollziehen und sich vor dem Transfer auf den Inhalt des Datenstroms verständigen und auf ein Format einigen. Deshalb wird der TCP Stream als **Nicht strukturierter Stream** kategorisiert.

TCP Verbindungen sind **Vollduplex Verbindungen**. So sind zwei voneinander getrennte Ströme innerhalb einer Verbindung möglich, die jeweils Daten in entgegengesetzter Richtung transportieren können. So kann eine Protokollsoftware Steuerungsinformationen in eine Richtung übertragen, während Daten in die andere gesendet werden. Dadurch wird das Datenaufkommen im Netz reduziert.

2.4.2.2 3 – Wege - Handshake

Um eine verlässliche Verbindung zu gewährleisten, wird beim Transmission Control Protokoll eine Technik verwendet, die beide Seiten der Kommunikation ständig dazu veranlasst, den Empfang von Informationen zu bestätigen. Erfolgt eine Bestätigung, kann der Vorgang fortgesetzt werden, andernfalls wird er abgebrochen oder nach einer von einem Timer gesetzten Zeit wiederholt. Sehen wir uns dazu den Aufbau einer TCP Verbindung an.

Dabei wird ein 3-Wege-Handshake verwendet (siehe Abbildung 17). Dazu wählt der Sender, in unserem Fall der Client, eine Sequenznummer (SEQ = x) an den Empfänger, der hier der Server ist, und setzt das SYN Flag, ein Bit, womit er dem Server zeigt, dass er eine Verbindung synchronisieren will.

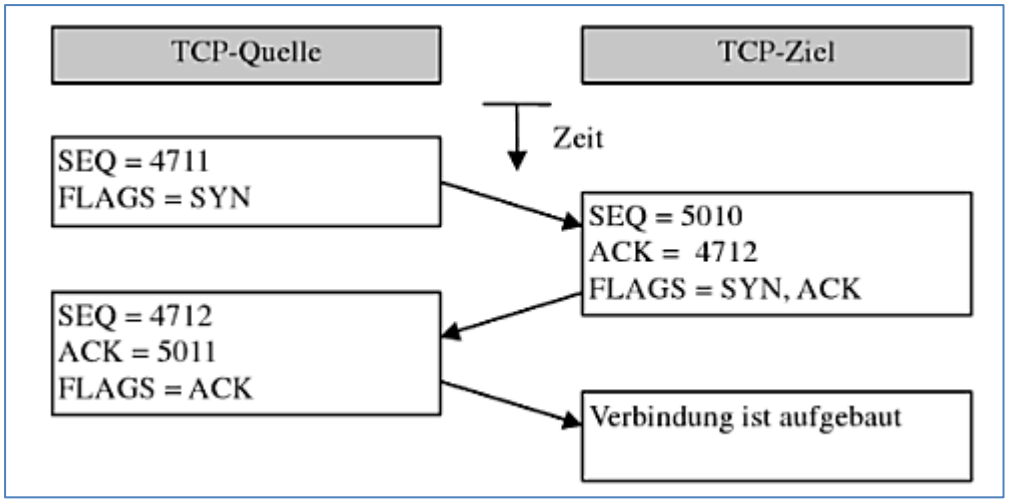

Abbildung 20: Der Aufbau einer TCP Verbindung wird über einen 3-Wege-Handshake realisiert. [96]

Nimmt der Server die Anfrage an, sendet er seine eigene Sequenznummer an den Client und sendet die Flags SYN und ACK. Mit ACK akzeptiert er die Anfrage, mit SYN stellt er eine Synchronisierungsanforderung. Jetzt verfügen Client und Server über beide Sequenznummern, worauf die Verbindung hergestellt werden kann. Der Client schickt sein ACK Flag an den Server und die Verbindung ist aufgebaut.

Zum Abbau der Verbindung nach dem Datentransfer wird beidseitig jeweils getrennt beendet. Dazu werden vier TCP-Segmente ausgetauscht. Beenden kann die Verbindung entweder der Client oder der Server. Dazu wird die Sequenznummer statt mit SYN Flag mit dem FIN Flag gesendet. Dies wird von der Gegenseite mit ACK bestätigt. Danach sendet die Gegenseite seinerseits noch ein FIN Flag, was ihm wiederum bestätigt wird (ACK). Wird das ACK der Gegenseite im selben Paket mit dem FIN Flag versendet, spricht man wie beim Aufbau der Verbindung von einem 3-Wege-Handshake.

2.4.2.3 Fazit Transmission Control Protokoll

Der Vorteil des Transmission Control Protokolls gegenüber anderen wie dem User Datagram Protokoll (UDP) liegt in der zuverlässigen Verbindung. Es wird dafür gesorgt, dass Daten bei Fehlern erneut gesendet werden, wenn eine Fehlerkorrektur versagt hat. Das Buffering sorgt dafür, dass fragmentierte Datensegmente in der richtigen Reihenfolge ankommen und nicht später gesendete vor den vorher gesendeten, wie es bei UDP vorkommt. Dadurch kann es allerdings zu Verzögerungen kommen. Außerdem muss beachtet werden, dass an jedes TCP

[96] Bild: Stein, Taschenbuch Rechnernetze und Internet, 2008, Seite 361, Bild 9.12

Segment ein Header von mindestens 40 Byte angefügt wird, der bei kleineren Datenpaketen verhältnismäßig viel Speicher beansprucht, so dass sich der Nutzen eher gering hält.

Für unsere Zwecke ist TCP sehr gut geeignet, da es sich bei den Updates vorzugsweise um größere Dateien von einigen Megabyte handelt, die sicher ankommen müssen. Fehlerhafte Übertragungen könnten schwerwiegende Programmfehler auslösen und eine Fehlersuche schwierig gestalten.

2.4.3 Windows Manifeste und Unattended Update Installation

2.4.3.1 Windows Manifeste integrieren

Wie bereits in den Anforderungen[97] beschrieben ist es in Netzwerken, also Client-Server Systemen unüblich, den Clients Rechte zur Administrierung ihrer PCs zu erteilen. Mangels Rechten beim Schreiben von Dateien kann es hier zur Verweigerung des Betriebssystems führen, Dateien zu kopieren oder zu überschreiben. Um dem entgegenzuwirken können mehrere Herangehensweisen betrachtet werden. Die einfachste Methode wäre es, den Administrator zu den Client-PCs zu schicken und dort jedes Update mit Administratorrechten ausführen zu lassen. Dazu kann man eine Windows Manifest Datei zum Programm hinzufügen oder integrieren. Manifest Dateien sind XML Dateien, die dem Betriebssystem Informationen über eine Datei geben, die gestartet werden soll. Das können EXE und DLL Dateien sein. Manifest Dateien haben die Endung .manifest und werden direkt derjenigen Datei zugeordnet, über die sie Informationen enthalten sollen. Sie heißen zum Beispiel *MeineAnwendung.exe.manifest*. Die XML Struktur der Manifest Datei beinhaltet Informationen über die Prozessorarchitektur, für die die Datei erstellt wurde, den Dateinamen, die Sprache und unter mehreren anderen Möglichkeiten, wie die Auswahl eines Windows Stils durch das Nachladen einer DLL, auch Informationen über die Sicherheitsanforderungen der Datei. [98]

Um Administratorrechte anzufordern, wird hier das Attribut **requireAdministrator** eingetragen. Öffnet das Betriebssystem die Datei, liest es dieses Attribut und startet als Konsequenz das UAC, das den User auffordert, sich als Administrator auszugeben. Kann er das bestätigen, erhält die Anwendung volle Zugriffsrechte auf administrativer Ebene.

[97] Vgl.: Kapitel 3.1.1 Kundenanforderungen
[98] Vgl.: http://www.samlogic.net/articles/manifest.htm

Diese Methode ist sicherlich für sehr kleine Netzwerke oder einzelne PCs gut geeignet, für größere Netzwerke ist sie nicht zu bevorzugen, da hier der Aufwand für den Administrator proportional zur Anzahl der Clients ansteigt.

2.4.3.2 Updates unattended einspielen

Wie bereits erläutert[99] ist ein Unattended Setup eine weitgehend automatisierte Installation. Wird sie mit dem Parameter */quiet* aufgerufen, läuft sie im *Silent* Modus, also komplett ohne GUI. Übertragen auf Updates bedeutet diese Herangehensweise, dass automatisch im Hintergrund nach Updates gesucht wird und diese installiert werden. Das kann auch mit grafischen Oberflächen auf Wunsch ergänzt werden, wie es das Windows Update zeigt.[100]

Möchte man Updates bequem ohne GUI, also Silent, einspielen, reicht es für den Zweck aus, die Oberfläche des Update Programms beim Start zu verstecken und alle Nutzereingaben zu simulieren. Hier besteht eine Ähnlichkeit zur skriptgesteuerten Installation[101], bei der Nutzereingaben vorher per Skript bestimmt werden, welches dann während der Installation ausgeführt wird. Abstrakt bedeutet das, man schreibt dem Programm vor, was es in einer Reihenfolge abzuarbeiten hat (Vgl. Abbildung 18).

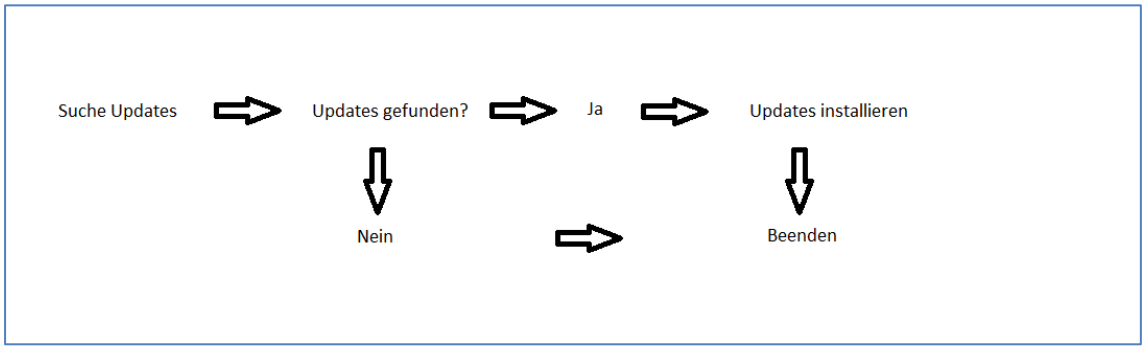

Abbildung 21: Schematische Darstellung einer Updateprozedur.

Hat man weitergehende Anforderungen an eine Silent Installation, wie die Update Suche und Installation zu bestimmten Zeitpunkten oder regelmäßigen Intervallen, reicht der programmatische Aufruf eines Update Installers nicht mehr aus. Schließlich kann man nicht vom Kunden verlangen, das Programm ständig geöffnet zu haben oder es zu gewissen Zeiten öffnen zu müssen. Abhilfe schafft hier eine Technik, die im folgenden Kapitel beschrieben werden soll.

[99] Vgl.: Kapitel 1.3.2 Unattended Setup und Skriptgesteuerte Installation
[100] Vgl.: Kapitel 1.7 Das Windows Update
[101] Vgl.: Kapitel 1.3.2 Unattended Setup und Skriptgesteuerte Installation

2.4.4 Registrierung als Windows Dienst

Um modernen Anforderungen an Updates gerecht zu werden, ist es vorteilhaft, ein Updatetool als Dienst für Windows bereitzustellen. Um einen Dienst bei Windows anzumelden, startet man in einer mit Administratorrechten gestarteten Eingabeaufforderung oder Powershell die Dienstdatei, die als EXE vorliegen muss, mit dem Parameter **–install**. Zum Abmelden wird der Parameter **–uninstall** verwendet. Gestartet wird der Dienst dann mit **net start Dienstname**, gestoppt wird er mit **net stop Dienstname**.

Der Dienst kann nun auf Wunsch bei jeder Nutzeranmeldung am Betriebssystem mit gestartet werden, läuft also ständig im Hintergrund. Mit einem Timer versehen, kann er nun in bestimmten Abständen nach Updates suchen und sie gegebenenfalls installieren. Da der Dienst mit Administratorrechten installiert wurde, erhält er auch zur Laufzeit die nötigen Zugriffsrechte für die Installationen.

3 Entwicklung des Update Tools

3.1 Entwicklungsumgebung

Das Update Tool wurde mit der RAD (Rapid Application Development) Umgebung XE2 von Embarcadero[102] entwickelt. Dazu wurde die Programmiersprache Delphi verwendet. RAD Studio benutzt einen WYSISWYG[103] Editor für die Oberfläche, um zügig grafische Elemente zu erstellen. Das ab Version XE2 enthaltene Framework Firemonkey[104] bietet mit seinen GPU unterstützten, vektorbasierten HD UI-Komponenten, die mit programmierten Effekten und Animationen aufwarten, eine optisch sehr ansprechende Oberfläche für Programme, die damit außer für Windows x32 und x64 auch noch für das MAC OS und iOS kompiliert werden können.

Die Vorteile von Delphi, dessen Vorgänger Turbo Pascal bereits im November 1983 erschien, liegen im Kompromiss aus Einfachheit und Komplexität. Die Sprache ist sehr umfangreich und doch leicht zu erlernen. Sie bietet alle Möglichkeiten der Objektorientierten Programmierung und kann durch die Komponenten des RAD Studios mit der Konkurrenz mithalten, nicht zuletzt aufgrund der Möglichkeiten zur Erweiterung der Komponenten. Es können eigene Komponenten erstellt und mit anderen Entwicklern geteilt werden. Viele Lösungen wurden schon gefunden und man kann diese, auch wenn oft nur kommerziell erhältlich, schnell in eigene Projekte integrieren und nutzen. Dazu läuft die IDE stabil und bietet viel Komfort. Dazu benötigen Delphi Anwendungen keine Runtime und je nach Anwendung auch keine Installer, da oft die Auslieferung der EXE Datei genügt.

Die Wahl für Delphi fiel daher nicht schwer, ebenfalls bedingt durch die Neuerungen, die Firemonkey in die Entwicklung brachte. Die Möglichkeit jeder Komponente Stile zuzuweisen und Animationen zu erzeugen, sogar dreidimensionale Anwendungen zu entwickeln, kann sicherlich als Spielerei abgetan werden, jedoch entsteht hier eine Technologie, die zukunftsfähig sein könnte.

[102] Vgl.: http://www.embarcadero.com/
[103] WYSISWYG: What you see is what you get
[104] Vgl.: http://www.embarcadero.com/de/products/firemonkey

3.2 Komponenten

Wie im vorhergehenden Kapitel bereits erwähnt, bietet das RAD Studio zahlreiche Komponenten zur Entwicklung von Programmen. Das erleichtert die Arbeit enorm und spart viel Zeit. Dazu können problemlos Komponenten von Drittanbietern in die eigenen Projekte integriert werden.

3.2.1 Die Indy Komponenten

Den Kern einer Applikation zur Softwareaktualisierung, die hauptsächlich über das Internet kommuniziert, bildet die Komponente, die über ein Netzwerk mit einem Server kommunizieren kann. Diese Möglichkeit bieten die Open Source Komponenten des Indy Projekts.[105]

Das Indy Projekt wurde ins Leben gerufen, um es Anwendungen zu ermöglichen, in Netzwerken zu kommunizieren. Dazu steht eine umfangreiche Socket Bibliothek zur Verfügung.

Unter den Indy Komponenten befinden sich FTP-, TCP-, Pop3-, http-, DNS-Server und viele mehr. Ebenfalls enthalten sind die entsprechenden Clients sowie Telnet, Whois oder Proxy Komponenten. (Vgl. Abbildung 19)

[105] Vgl.: http://www.indyproject.org/index.de.aspx

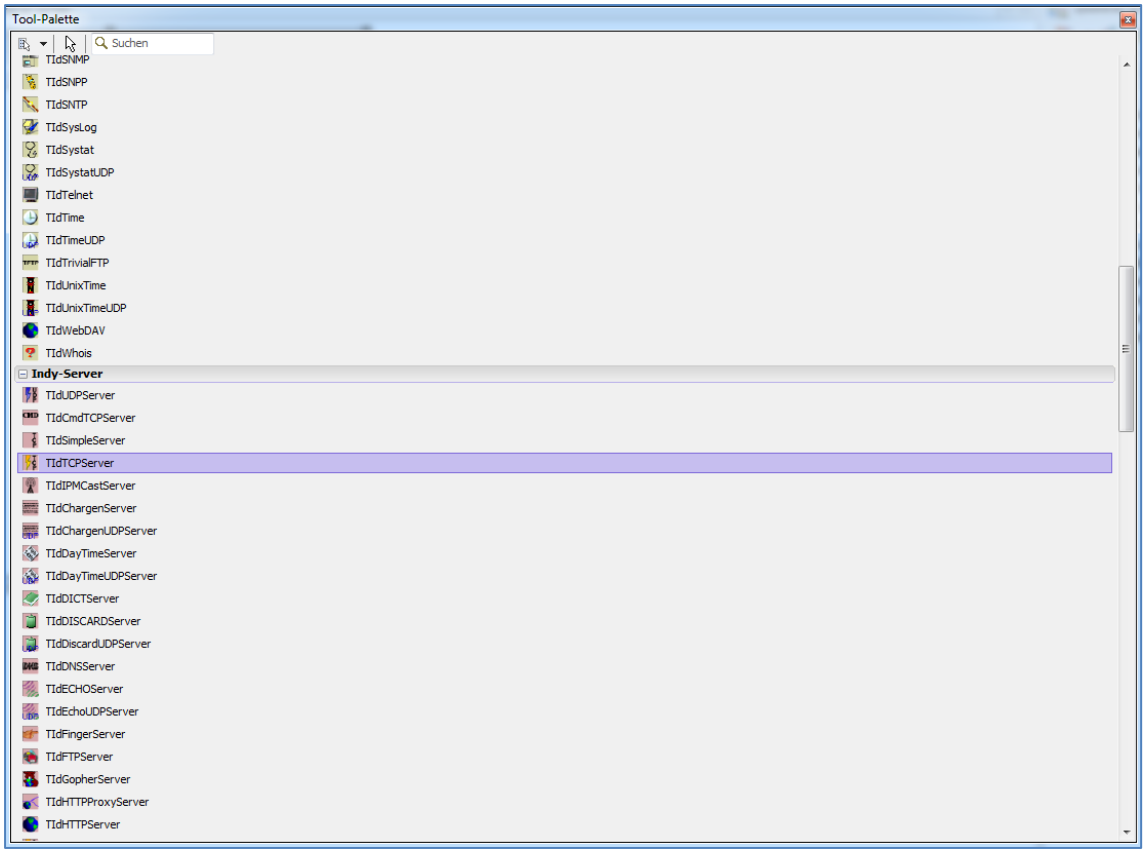

Abbildung 22: Einige Indy Komponenten aus der Tool Palette des RAD Studios

Die für das Projekt Update wichtigsten Komponenten sind hier **TidTCPServer** und **TidTCPClient**. Sie kommunizieren im Netzwerk miteinander und transferieren die Updatedateien vom Server zum Client.

Die TidTCPServer Komponente implementiert einen multithread fähigen TCP Server. Sie erlaubt mehrere Listener Threads, die auf Verbindungsversuche von Clients warten und diese akzeptieren können. Es können mehrere Verbindungen gleichzeitig aufrechterhalten werden, da für jeden Client ein neuer **TidContext** als Task erzeugt wird. Diese werden über einen Scheduler verwaltet, der die Tasks erzeugt, ausführt und beendet. Bereits beim OnConnect Ereignis des TidTCPServer, welches durch einen erfolgreichen Verbindungsaufbau aufgerufen wird, wird ein Kontext übergeben, der bis zum Verbindungsabbau für genau einen Client zugewiesen ist und für den kompletten Informationsaustausch zwischen Server und Client als Container arbeitet. So können problemlos mehrere Kontexte parallel agieren, wobei darauf geachtet werden muss,

dass die Anwendung in ihrer Gesamtheit threadsicher entwickelt wird, da es sonst zu Deadlocks[106] kommen kann.

3.2.2 ActiveX Data Objects

ActiveX Data Objects (ADO) ist eine Windows Schnittstelle zum Aufbau von Datenbankverbindungen und zum Zugriff auf Datenbanken. Über eine Connection Komponente wird mittels Connectionstring, eine Folge von Zeichen, eine Verbindung zu einer Datenquelle wie Access oder SQL Datenbanken eingerichtet. Dann kann über ein Dataset eine SQL Abfrage ein Ergebnis liefern und an das Programm senden. ADO arbeitet zügig, verbraucht wenig Speicher und kann Datenbankverbindungen mit mehreren Providern wie OLEDB, Jet oder MSDataShape aufbauen. Die Komponenten sind gut für jegliche Datenbankabfragen geeignet. Für das Update Tool werden **TADODatSet** und **TADOConnection** verwendet.

3.2.3 AbZipper

Für das Verpacken und Entpacken von Dateien in das gängige Zip Format wird die Drittanbieter Komponente AbZipper verwendet. Sie wird für die Möglichkeit zum Offline-Update verwendet. Hat ein Client PC keinen Internetzugang, kann er so Updates entweder von der Webseite des Unternehmens oder gepackt von einem anderen PC mit Internetzugang und installiertem Update Tool erhalten. An letzterem kann er das Update herunterladen und verpacken, danach einfach an seinem Rechner entpacken und installieren lassen. Zum Verpacken wird TAbzipper, zum Entpacken die TAbUnZipper Komponente benötigt.

Abzipper beherrscht mehrere Kompressionsmethoden und kann Zip-Dateien mit Passwort und Kommentaren erstellen. Er ist für die Anforderungen durch sein einfaches Handling und die mehr als akzeptable Geschwindigkeit gut geeignet.

[106] Deadlock: Vgl.: Schneider, Taschenbuch der Informatik, 2007, S. 279

4 Realisierung der Anforderungen

Um ein Tool zum Update von Softwareanwendungen bereitzustellen, werden server- und clientseitige Anwendungen benötigt, die miteinander kommunizieren können. Daher war es praktisch, innerhalb einer Projektgruppe ein Projekt jeweils für die serverseitige und clientseitige Anwendung anzulegen. Die Projekte können unabhängig voneinander kompiliert und debuggt werden. Zusätzlich wurde ein Projekt für den Windows Dienst zur Installation von Updates erstellt, welches Gemeinsamkeiten mit dem Update Client Projekt haben sollte, da es als Update Dienst ähnlich arbeiten sollte wie die Desktop Anwendung.

In den folgenden Kapiteln werden die einzelnen Projekte in ihrem Aufbau und der Funktionsweise erläutert. Beachten Sie dabei bitte, dass sich die Projekte, während diese Diplomarbeit verfasst wurde, noch in der Entwicklungsphase befanden, sodass es noch zu Änderungen in den finalen Kompilaten kam. Nicht auszuschließen ist auch eine Änderung nach dem Release, da Software in ihrem Lebenszyklus [107]immer wieder verbessert und ausgebessert werden muss, speziell wenn es sich um Software handelt, die dafür konzipiert wurde. Für den Fall, dass das Update Tool selbst aktualisiert werden soll, wurde ein Projekt außerhalb der Projektgruppe erstellt, welches im Anschluss an die Erläuterungen der Projektgruppe erklärt wird.

4.1 Voraussetzungen

Um die Anwendungen für Client und Server nutzen zu können, müssen auf der jeweiligen Seite bestimmte Voraussetzungen erfüllt sein.

4.1.1 Clientseitige Voraussetzungen

Das Updatetool der *Vordruckverlag Weise GmbH* ist speziell auf die Bedürfnisse des Unternehmens und der Kundschaft abgestimmt und ist nicht für die Weitergabe an Dritte vorgesehen. Ob sich diese Entscheidung des Managements später differenziert, ist nicht Thema dieser Arbeit und wird bisher programmtechnisch nicht unterstützt. Im Gegenteil werden programmtechnische Sperren errichtet, die dafür sorgen, dass nur VVW Anwendungen erkannt werden.

[107] Vgl.: Kapitel 1.1.2 Überblick der Update Varianten

Damit ist die erste Voraussetzung zur clientseitigen Nutzung des Tools das Vorhandensein von Software der *Vordruckverlag Weise GmbH*. Die Software trägt sich bei der Installation in die Windows-Registrierung des Nutzers ein und wird vom Updatetool auch nur an der bestimmten Stelle erkannt. Ist keine Software installiert, können auch keine Updates gefunden werden.

Die zweite Voraussetzung zur Nutzung des Updatetools ist eine gewichtige. Der Nutzer benötigt für das Kopieren und Überschreiben von Dateien in seinem System unbedingt Schreibrechte, deshalb fragt das Tool direkt beim Programmstart nach Administratorrechten für das Betriebssystem. Um das zu umgehen, weil es etwa bei Client-Server Versionen der VVW Software nicht üblich ist, den Clients die Rechte zu erteilen, kann auf die Dienstanwendung zurückgegriffen werden. Ob die Dienstanwendung für jeden Client oder nur für Client-Server Anwendungen deployt wird, stand zum Zeitpunkt der Anfertigung dieser Arbeit noch nicht fest.

Sind diese Voraussetzungen erfüllt, kann das Tool zur Aktualisierung der VVW Software genutzt werden.

4.1.2 Serverseitige Anforderungen

Um das Tool erfolgreich nutzen zu können, muss die serverseitige Anwendung auf dem Webserver der VVW GmbH installiert und korrekt eingerichtet werden. Die Konfiguration des Server Tools wird in den folgenden Kapiteln erläutert.

Jedoch sind noch einige Dinge für den Webserver zu beachten. Die Dateien, die für Updates bereit gestellt werden sollen, müssen in korrekten Ordnern in den richtigen Unterordnern liegen, da die Dateisysteme vom Updatetool nicht analysiert sondern ausgelesen und übernommen werden. Es könnte sonst zu Programmfehlern oder falscher Adressierung führen.

Um die gewünschte Lizenzerkennung durchführen zu können, wird eine Datenbank auf dem Webserver benötigt, die dem Tool Auskunft darüber erteilen kann, welcher Kunde über welche Software verfügt und in welcher Version diese vorliegt. Daran kann das Server Tool dann erkennen, welche Updates dem Nutzer zur Verfügung gestellt werden sollen.

Die Installation einer Kundendatenbank auf dem Webserver ist sicherlich eine bedenkliche Entscheidung, da hier im Fall von Angriffen Passwörter der Kunden gestohlen werden könnten. Daher werden spezielle Sicherheitsmaßnahmen getroffen werden müssen, die hier nicht weiter thematisiert werden sollen.

4.2 Erläuterung der serverseitigen Anwendung

Der Arbeitstitel der serverseitigen Anwendung lautet **UpdateServer.exe**. Im Folgenden wird die Anwendung als UpdateServer oder Server bezeichnet. Diese Begriffe beziehen sich dann auf die genannte Anwendung.

Der UpdateServer soll eine schlanke Anwendung im Consolenstil sein, die auf Clientanfragen reagieren und Dateien versenden kann. Die Anwendung besteht aus visuellen und nichtvisuellen Komponenten, die in den folgenden beiden Kapiteln beschrieben werden.

4.2.1 Visuelle Komponenten

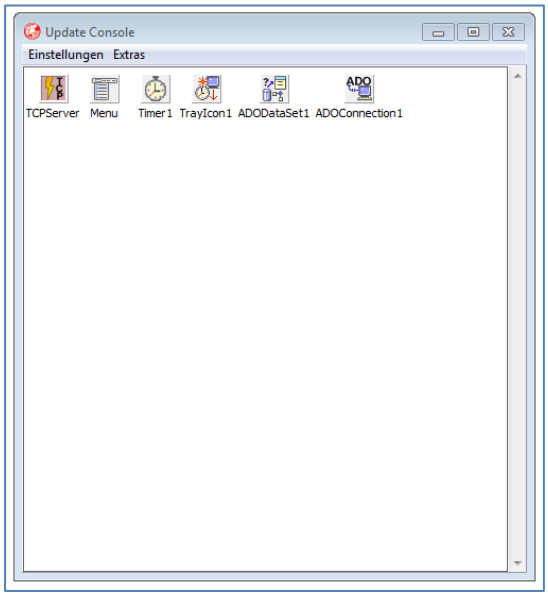

Abbildung 23: UpdateServer zur Entwicklungszeit

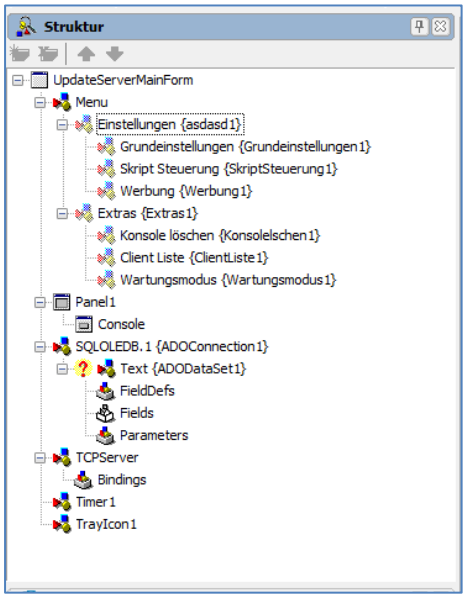

Abbildung 24: Strukturansicht des Updateservers

4.2.1.1 Das Haupt Formular

Die Oberfläche des UpdateServers besteht im Wesentlichen aus einer Menü Zeile und einem Textfenster (Vgl. Abbildung 20). In der Strukturansicht (Vgl. Abbildung 21) erkennt man die Menüeinträge *Grundeinstellungen*, *Skript Steuerung* und *Werbung* unter dem Menüreiter *Einstellungen* sowie die Einträge *Konsole löschen*, *Client Liste* und *Wartungsmodus* unter dem Reiter *Extras*.

Auf einem Panel liegt die Komponente *Console* vom Typ *TMemo*. Die Konsole soll zur Laufzeit wiedergeben, welcher Client sich mit dem Server verbinden will und enthält die wesentlichen Informationen zur Kommunikation zwischen Client und Server. Diese Informationen werden zusätzlich in einer Textdatei gespeichert.

Eine weitere visuelle Komponente ist das *TrayIcon*. Bei der Auslösung des *Minimieren* Events des Formulars wird die Anwendung unsichtbar und ein Symbol im System Tray erzeugt, welches die Funktionalität besitzt, das Programm wieder anzuzeigen oder zu beenden.

4.2.1.2 Die Einstellungsmöglichkeiten

Über den Menüpunkt *Grundeinstellungen* gelangt man zu einem Formular, das einige Einstellungsmöglichkeiten zulässt. Die Einstellungen sollten im Wartungsmodus vorgenommen werden, der automatisch Clients beim Verbindungsversuch ablehnt und ihnen mitteilt, dass gerade eine Wartung stattfindet. In dem Formular können zunächst die Ordner der Dateien angegeben werden, die für Updates bereitgestellt werden. Es wird ein Ordner für die generellen Dateien und ein separater für die Dateien mit Servicepaketfunktionalität angegeben. Anschließend wird noch das Verzeichnis für gemeinsam genutzte Dateien erfragt, welcher von mehreren Programmen verwendet werden soll. Sind diese Ordner nicht angegeben, wird das Programm nie Dateien zum Update finden und damit entscheidend an Leistungsfähigkeit einbüßen. Es wird unbrauchbar.

Als weitere Einstellungsmöglichkeiten kann der Administrator eine maximale Verbindungsanzahl angeben und Benutzername sowie Passwort für die Kundendatenbank eingeben.

Die *Skript-Steuerung* ist ein Feature, das dem Verteiler von Updates erlaubt, vor oder nach der Installation der Aktualisierungen Skripte oder Anwendungen zu starten. Dies kann notwendig sein, wenn zum Update zusätzliche Einstellungen für spezielle Programme vorgenommen werden müssen, die nur über Skripte oder kleine Anwendungen realisiert werden können, wie

etwa Datenbank Tweaks oder den Austausch von Formularen innerhalb von Tabellen. Damit können natürlich auch Dateien wie PDF- oder Text-Dateien nach dem Update geöffnet werden, um Anweisungen oder Informationen weiterzugeben.

Im Menüpunkt *Werbung* kann ein Text, zum Beispiel zu Werbezwecken, angegeben werden, der während des Updates beim Nutzer erscheint und ihn auf Neuerungen oder Angebote hinweist.

Die Punkte unter dem Reiter *Extras* sind kleine Funktionen enthalten, die den Inhalt der Console löschen, die aktuell am Server angemeldeten Clients ausgeben und den Wartungsmodus an- oder ausschalten können.

4.2.2 Nichtvisuelle Komponenten

Die nichtvisuellen Komponenten der Server Anwendung bestehen aus einer *TADOConnection*, einem *TADODataSet* und einem *TidTCPServer*. TADOConnection ist eine Komponente, die wie der Name schon vermuten lässt, über die ADO Schnittstelle eine Verbindung zu einer Datenbank aufbauen kann.[108] Dazu wird ein Connectionstring benötigt, der den Provider, in diesem Fall ein OLE DB Provider für SQL Server, eine Information, ob die integrierte Sicherheit von Windows NT oder Benutzername und Passwort verwendet wird, sowie den Datenbankennamen und den Ort der Datenbank enthält.

Ist eine Verbindung mit der Datenbank erfolgreich hergestellt worden, kann über das DataSet auf die einzelnen Tabellen der Datenbank zugegriffen und SQL Befehle gesendet werden. Für die Verknüpfung von DataSet und Datenbank muss bei der Eigenschaft *Connection* des DataSets lediglich die ADOConnection angegeben werden, die die für die Verbindung nötigen Eigenschaften an das DataSet weitergibt. Kennt das DataSet die Verbindung zur Datenbank, kann über die Eigenschaft *Command* eine SQL Abfrage eingegeben werden, die wiederum über das Setzen der Eigenschaft *Active* ausgeführt wird und alle Ergebnisse in einer Property *RecordSet* des Typs *TCustomADODataSet* sammelt.

Der *TCPServer* vom Typ *TidTCPServer* nimmt in der Anwendung eine zentrale Position ein. Er ist dafür zuständig, auf einem festgelegten Port zu lauschen und auf Client Anfragen zu reagieren. Er kommuniziert mit den Clients und erstellt für jeden Client einen eigenen Context.[109] Über ihn kann mit dem TCPClient des Clients interagiert werden und der Server kann diesen nach

[108] Vgl.: Kapitel 4.2.2 ActiveX Data Objects
[109] Vgl.: Kapitel 4.2.1 Die Indy Komponenten

Informationen befragen. Außerdem ist er für den Dateitransfer zwischen Server und Client verantwortlich.

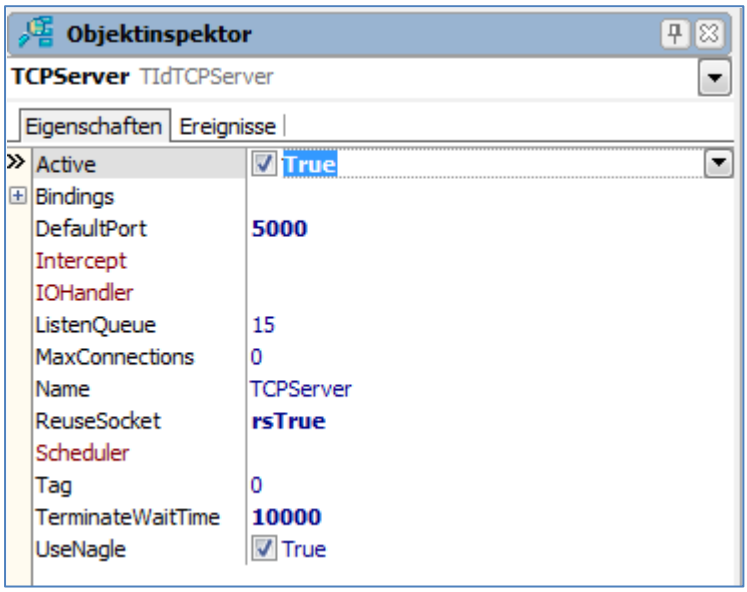

Abbildung 25: Die Eigenschaften des TCPServers im Objekinspektor

Im Objektinspektor werden die Eigenschaften des TCPServers eingerichtet. Die Eigenschaft *active* aktiviert den Server, sodass er auf dem eingestellten *DefaultPort* lauschen kann. Das ist der Port, an dem die Clients ihn erreichen können.

Der Wert der *ListenQueue* gibt an, wie viele unbeantwortete Verbindungsversuche für jeden Listener Thread akzeptiert werden. Wenn Listener Thread mehr unbeantwortete Anfragen erhält, als der in ListenQueue angegebene Wert zulässt, beendet der TCPServer die Verbindung.

ReuseSocket gibt an, ob die Socket IP Adresse oder Port Nummer von Listener Thread Instanzen des Servers genutzt werden können, also, ob der Socket nach dem Schließen einer Verbindung wiederverwendet werden kann.

TerminateWaitTime gibt an, wie lange der Server beim Beenden warten soll, bis alle Verbindungen beendet wurden. Genauer gesagt prüft er alle 250ms die Länge der Context Thread Liste. Wenn die TerminateWaitTime vorbei ist, bevor die Context Thread Liste leer ist, also die Länge 0 hat, wird eine Exception ausgelöst.

UseNagle bestimmt, ob der Nagle Algorithmus benutzt werden soll. Dieser ist dafür zuständig, die Sendung von sehr kleinen Paketen aufzuhalten, da damit der Durchsatz der Verbindung verringert werden könnte. So werden erst große Pakete gesendet, die kleinen Pakete am Ende.

Im Hintergrund arbeitet *TidScheduler*, der die Verwaltung der Tasks des TCPServers übernimmt. So wird für jede Verbindung ein eigener Task erstellt, für den Methoden zur Erstellung,

Ausführung und Beendigung verfügbar sind. Zur Erstellung der Tasks wird die Klasse *TidServerContextClass* zur Hilfe genutzt. Bei der Ausführung der Listener Threads wird ein TidContext erstellt, der in der Contexts Property, einer Thread Liste, gespeichert wird. Durch diese Funktionalität ist der TCPServer multi-thread fähig und kann Anfragen von verschiedenen Clients gleichzeitig abarbeiten.

4.2.3 Arbeitsweise des Servers

Ist der Server korrekt konfiguriert, kann er als möglichst ständig laufende Anwendung auf Verbindungsversuche von Clients am zugewiesenen Port lauschen. Verbindet sich ein Client erfolgreich, eröffnet der TCPServer einen neuen Context für den Client und wartet auf spezielle Kommandos. Diese werden in der TCP Verbindung über den IO Controller des zuständigen Contexts in einem String übermittelt. So kann sichergestellt werden, dass die Kommunikation zwischen Client und Server nicht von anderen Clients gestört werden kann.

Die wichtigsten Eckpunkte der Serveranwendung sind die Ermittlung der Daten eines Kunden über den Zugang zur Kundendatenbank, die Durchsuchung von Ordnern nach Dateien und Sammlung von Informationen über die Dateien sowie der Versand der Dateien zum Client. Das Ganze sollte parallel für mehrere Clients möglich sein, weshalb eine spezielle Anforderung an die Threadsicherheit des gesamten programmtechnischen Ablaufs gerichtet ist.

Zur Überprüfung der Kundenummer des Clients erstellt der Server über die in Kapitel 5.2.2 genannte TADOConnection eine SQL Abfrage, die in der Kundendatenbank mehrere Informationen über die zu der Kundennummer zugehörigen Daten zurückgibt. Die für den Server interessanten Daten sind die Softwaremodule, die der jeweiligen Kundennummer zugeordnet sind, also die Software, die der Kunde käuflich erworben hat. Diese Daten bestehen aus dem Namen der Software, der Versionsnummer, der Information, ob ein Servicepaket für die jeweilige Software erworben wurde und der Laufzeit des Servicepakets. Mit diesen Daten kann der Server nun entscheiden, ob der Kunde speziellen Content zum Download erhalten soll oder nicht. Denn nur Kunden mit gültigem Servicevertrag erhalten zusätzliche Downloads für ihre Software und müssen vor dem Download ein Passwort für die Servicepaketupdates eingeben. Die üblichen Downloads, also Hotfixes, Patches und weitere Updates benötigen hingegen keine Einträge auf der Kundendatenbank. So sind auch für Nutzer von Demoversionen sämtliche Standardinhalte frei verfügbar. Die Restriktionen der Demoversionen sind innerhalb der einzelnen Programme integriert und somit für den Updateserver nicht relevant. So können auch

Updates für Softwareanwendungen runtergeladen werden, die nicht in der Kundendatenbank für den Nutzer registriert sind.

Die Durchsuchung von Ordnern wird mittels eines *TSearchRec* bewältigt. Es liefert die Dateiinformationen über eine Datei, die mittels *FindFirst* und *FindNext* Methoden rekursiv gesucht werden können. Wird eine Datei gefunden, werden die Felder des TSearchRec-Parameters modifiziert, um die gefundene Datei zu bestimmen.

4.3 Erläuterung der Clientseitigen Anwendung

Die Client Anwendung mit dem Arbeitstitel **UpdateClient.exe** ist eine Firemonkey HD Desktop Anwendung. Sie wird im Folgenden als UpdateClient oder Client bezeichnet.

Die Anwendung besteht aus visuellen und nichtvisuellen Komponenten, die in den beiden folgenden Kapiteln kurz erläutert werden sollen.

4.3.1 Visuelle Komponenten

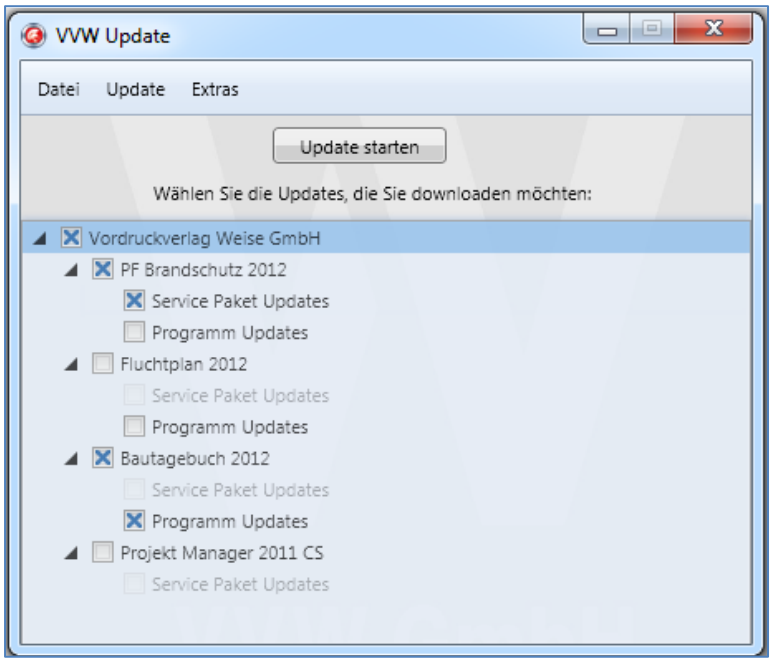

Abbildung 26: Der UpdateClient zur Laufzeit

Die Oberfläche des Client Tools besteht aus einem Menü mit den Punkten *Datei*, *Update* und *Extras* und einer TreeView Komponente. Im TreeView kann der Nutzer auswählen, welche Produkte er aktualisieren will. Dabei bekommt er sämtliche Software angezeigt, die auf seinem

PC installiert wurde. Sind Servicepaketupdates verfügbar, wird der Unterknoten *Servicepaketudpates* freigeschalten, ansonsten ist der Punkt *Programmupdates* für alle normalen Updates öffentlich verfügbar. In Abbildung 23 kann man sehen, dass für *Projekt Manager 2011 CS* kein normales Update verfügbar ist, aber ein Servicepaketupdate ausgegraut dargestellt wird. Hier hat der Kunde bereits ein normales Update bezogen und verfügt nicht über die Rechte zu Servicepaketupdates. Die Möglichkeit für Updates wird trotzdem angezeigt, um den Kunden zu verdeutlichen, dass hier weiteres Material vorhanden ist, das ihm allerdings nicht zur Verfügung steht.

Der Nutzer kann im TreeView frei wählen, für welche Programme er Updates downloaden möchte. Die Anwendung speichert die letzte Auswahl und wählt beim nächsten Update die zuletzt gewählten Programme wieder an. Durch die Betätigung des *Update Starten* Buttons beginnt der Download für die gewünschten Programme.

4.3.1.1 Das Menü

Im Menü befinden sich unter dem Punkt *Datei* die Funktion zur Beendigung des Programms, unter dem Punkt *Update* Funktionen zum Bearbeiten der Einstellungen, zum Zurücksetzen der Updates und zum *Update aus Datei*. Der Menüpunkt Extras führt zum Aufruf eines Tools, welches das Updateprogramm selbst erneuern kann, falls eine neuere Version vorliegt.

Das Zurücksetzen des Updates hilft im Fall von Problemen oder Fehlern im Tool oder wenn ein Programm selbst Fehler verursacht. Dann wird der Datumsstempel aller Programme auf den ersten Stand zurückgesetzt und das Update kann alle auf dem Updateserver befindlichen Dateien erneut herunterladen. Diese Vorgehensweise wird wahrscheinlich bei beschädigten Dateien notwendig sein, die heruntergeladen wurden und irregulär arbeiten.

Über den Menüpunkt *Einstellungen* gelangt man in ein neues Formular, welches die Automatisierung von Updates regelt. Der Nutzer kann hier entscheiden, ob er Updates manuell oder automatisch installieren will. Bei letzterem kann er noch das Intervall und den Zeitpunkt für die Updateinstallation wählen (siehe Abbildung 24).

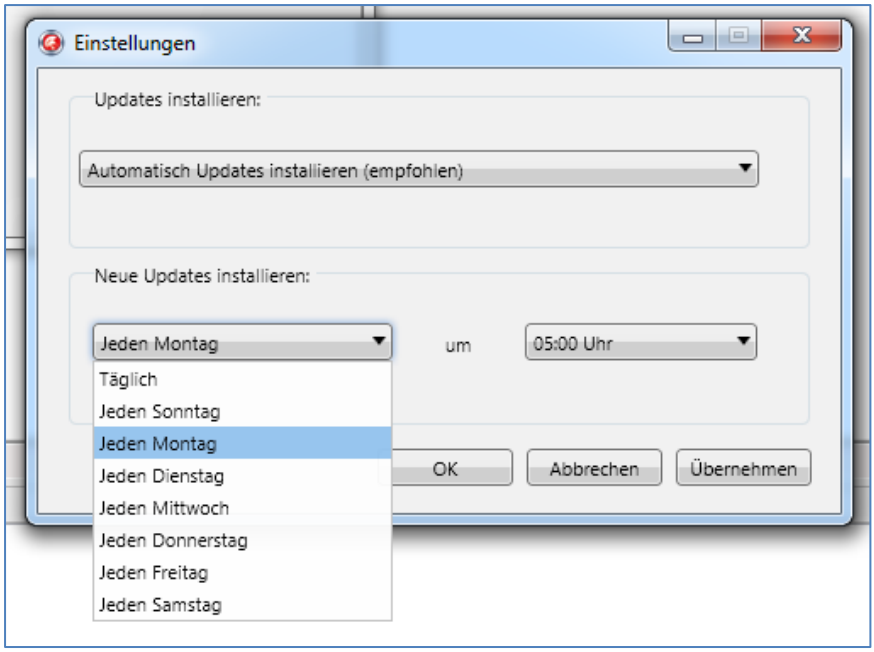

Abbildung 27: Der Nutzer kann den Intervall und die Zeit für automatische Updates einrichten

Update aus Datei, das über das *Updatemenü* aufgerufen wird, bietet Möglichkeiten zur Softwareaktualisierung ohne Internetzugang an bestimmten Arbeitsplätzen. Damit kann über eine Zip-Datei ein Update auf einem Rechner eingespielt werden, die vorher über die Homepage des Herstellers oder über das an einem internetfähigen PC installierte Updatetool bezogen werden kann. Letzteres ist nur für aktuelle Software möglich.

4.3.2 TidTCPClient

Der TidTCPClient der Anwendung ist die Indy TCP Komponente der Client-Seite, der eine Verbindung zum TCPServer aufbauen und mit diesem Informationen austauschen kann. Dazu muss er die IP Adresse des Hosts, also des Servers sowie den Port kennen, an dem gelauscht wird. Er stellt die Verbindungsanfrage und kommuniziert bei erfolgreichem Verbindungsaufbau über einen IOHandler mit der serverseitigen Anwendung.

4.4 Das Updatetool als Windows Dienst

Das Embarcadero RAD Studio bietet die Möglichkeit, beim Anlegen eines Projektes eine Serviceanwendung zu erstellen. Sie läuft als echter Windows Dienst im Hintergrund ohne Benutzeroberfläche. Er wird vom Administrator oder bei der Installation des Updatetools installiert und wird vom Betriebssystem bei der Nutzeranmeldung gestartet. Dann kann der Dienst in regelmäßigen Abständen nach Updates suchen und diese gegebenenfalls installieren.

Dazu muss der Dienst wie der Update Client einen TidTCPClient besitzen, mit dem er eine Verbindung zum TCPServer herstellen kann. Die wichtigsten Methoden wurden hierfür aus dem TCPClient Projekt referenziert, da sie bis auf Kleinigkeiten übereinstimmen. Von einem Projekt auf ein anderes zu verweisen ist nicht direkt möglich, weshalb eine Unit mit den gemeinsam verwendeten Methoden erstellt wurde, die in beide Projekte integriert wurde.

Prinzipiell arbeitet der Dienst genauso wie der Client, mit dem Unterschied, dass der Client die Programme nicht auswählen kann, die er aktualisieren möchte. Daher werden standardmäßig alle Programme immer auf aktuellstem Stand gehalten.

Das Intervall der Installation bestimmt ein Timer, der je nach Vorgabe den Haupttask des Dienstes startet und damit nach Updates suchen lässt. Der Timer lässt sich über das Menü Einstellungen in der Client Anwendung bearbeiten, die mit dem Dienst interagieren soll.[110]

4.5 Die Wechselwirkung oder Kommunikation von Client und Server

In Kapitel 4.2.3 *Arbeitsweise des Servers* wurde bereits beschrieben, wie der Server auf Clientanfragen reagiert. Die Arbeitsroutine des Servertools speziell im Umgang mit dem Client soll in diesem Kapitel thematisiert werden.

4.5.1 TCPServerExecute

Die TidTCPServer Komponente des Servertools führt im *TCPServerExecute* Ereignis die Methode *GetCommand* aus. *TCPServerExecute* wird immer aufgerufen, wenn ein verbundener Client aus dem idle Zustand heraus versucht, mit dem Server zu kommunizieren. Das Ereignis bekommt von der *TCPServer* Komponente den *TidContext* übergeben, welcher ebenfalls an die *GetCommand* Methode übergeben wird. Der Context ist für den weiteren Programmablauf sehr wichtig, da er alle Eigenschaften der Verbindung zu genau einem Client besitzt und wird deshalb jeder Methode übergeben, die mit dem Client kommunizieren wird. Zusätzlich wird noch ein String an *GetCommand* übergeben, der im *TCPServerExecute* Ereignis mit *AContext.Connection.IOHandler.ReadLn* abgefragt wird. *AContext* ist hier eine Instanz der *TidContext* Klasse, die nur für einen Client reserviert ist. Dessen Eigenschaft *Connection* gibt die zugrunde liegende Verbindung an. Im Allgemeinen ist die Verbindung eine Instanz von *TidTCPConnection*. Der *IOHandler* gibt die dieser Instanz zugeordnete Eingabe-/Ausgaberoutine

[110] Die Interaktion des Dienstes mit der Client Anwendung ist zur Zeit der Anfertigung dieser Arbeit noch in Planung.

an. *Readln* wartet auf eine Nachricht des Clients im String Format und gibt diesen als Ergebnis zurück.

4.5.2 GetCommand als Switch

Da die *GetCommand* Methode nun den Context und die Zeile der Clientanforderung kennt, kann sie weiter mit dem Client kommunizieren. Diese Zeile drückt die Anfrage des Clients aus. Dieser wird bei seinem Programmstart zunächst versuchen, eine Verbindung mit dem Server herzustellen und sobald diese steht nach Updates suchen wollen. Vorher liest er noch die auf dem PC des Clients installierten VVW Programme aus der Registrierung aus und speichert diese in einem Array.

Möchte der User ein Update „aus Datei“ machen, sendet er die Nachricht „Updates suchen_Off“, bei lokalen Updates „Updates suchen“ an den Server. Die *GetCommand* Methode, die diese Nachricht verarbeitet, kann nun durch Verzweigungen verschiedene Aktionen ausführen, die von den Anforderungen des Clients abhängen. Sehen wir uns den Fall „Updates suchen“ an: Sollen in den Ordnern des Servers neue Dateien gesucht werden, muss zunächst ermittelt werden, welche Programme für Updates überhaupt relevant sind. Dazu sendet der Server die Nachricht „Suche Produkte“ an den Client.[111] Nebenbei sei erwähnt, dass es eventuell üblicher ist, die Nachrichten, die zwischen Server und Clients gesendet werden, kryptischer zu gestalten. Aus Gründen der Anschaulichkeit sind diese aber leicht verständlich gewählt.

Erhält der Client die Nachricht, sendet er einzeln die Produktnamen der verschiedenen Anwendungen sowie das Datum des letzten Updates für jedes Produkt. Dieses Datum wird beim Client nach erfolgreichem Update in einer Initialisierungsdatei gespeichert. Hat er noch nie Updates bezogen, sendet er „0“, was die Konvertierungsfunktion *DateTimeToString* als 01.01.1899 zurückgibt. Mit den beiden Informationen kann die Serveranwendung nun mit einem *TSearchRec* die entsprechenden Ordner für Updatedateien durchsuchen und Dateien mit dem Datum des letzten Updates vergleichen. Existieren neue oder veränderte Dateien, schreibt er die Dateinamen in eine Dateiliste und merkt sich den dazugehörigen Programmnamen. Danach kann *GetCommand* eine Liste der Programme senden, für die neue Dateien zur Verfügung stehen. Danach tritt wieder ein idle-Zustand ein. Der Server wartet auf weitere Befehle.

[111] Vgl.: Abbildung 28

```
if (Befehl = 'Updates suchen') or (Befehl = 'Updates suchen_Off') then
begin
      if Befehl = 'Updates suchen_Off' then OfflineUpdate := true;
      FileCount := 0;
      SetLength(FileList, 0);
      WriteLn('Suche Produkte');
      KundenNummer := ReadLn;
      GetKundenData(KundenNummer);
      WriteLn(KundenName);
      Befehl := ReadLn;


      while Befehl <> 'Produkte gesendet' do
      begin
        LastUpdate := ReadLn();
        Logger.Write(AContext.Binding.PeerIP + ': ' + Befehl);
        if OfflineUpdate then Befehl := ProduktNameToOrdnerName(Befehl);
        LookForUpdates(Befehl, LastUpdate);  // Befehl hier : das Produkt; FileList füllen
        Befehl := ReadLn;
      end;
      if ReadLn() = 'Common' then
        LookForUpdates('Common', ReadLn());
```

Abbildung 28: Möchte der Client Updates suchen, wird er nach den Installierten Programmen gefragt sowie seiner Kundennummer befragt

Hat der User die Programme gewählt, die er aktualisieren will, sendet er an den Server die Nachricht „Update suchen“. Hier erfolgt die oben beschriebene Prozedur erneut, diesmal werden aber nicht alle Programme an den Server gesendet, sondern nur die im *TreeView* der Clientanwendung selektierten. Daraufhin erstellt die Serveranwendung eine neue Dateiliste.

Ist dieser Schritt erfolgt und wartet der Server wieder auf Nachrichten, sendet der Client „Update starten“. Dann kann der Server seine Dateiliste abarbeiten und dem Client die Dateien senden.

4.5.3 Senden der Dateien

```
procedure TUpdateServerMainForm.SendFile(FileName : string; Context : TIdContext; Comment : string);
var
Stream : TFileStream;
zipFileName : string;
begin
  try
    if not FileExists(FileName) then
    begin
      Console.Lines.Create.Add(Context.Binding.PeerIP + FileName + ' not found');
      Logger.write('ERROR: ' + TimeToStr(now) + Context.Binding.PeerIP + ': ' + ': ' + FileName + ' not found');
      Context.Connection.IOHandler.WriteLn(IntToStr(0));   //Stream hat Größe 0 -> Client weiß, dass keine Datei kommt
      Exit;
    end;
    zipFileName := ZipFile(filename, Comment);
    Stream := TFileStream.Create(zipFileName, fmOpenRead or fmShareDenyWrite);
    with Context do begin
      try
         Connection.IOHandler.WriteLn(IntToStr(Stream.Size));
         Console.Lines.Add(TimeToStr(now) + ': ' + 'Sende an ' + Context.Binding.PeerIP + ': ' + FileName + '(' + IntToStr(Stream.Size) + ' Byte
         Logger.Write('Sende an ' + Context.Binding.PeerIP + ': ' + FileName + '(' + IntToStr(Stream.Size) + ' Bytes)');
         Connection.IOHandler.WriteBufferOpen;
         Connection.IOHandler.Write(Stream);
      except
        on E:Exception do begin
          Console.Lines.Add(TimeToStr(now) + ': ' + E.ToString + Context.Binding.PeerIP);
          Logger.Write(TimeToStr(now) + ': ' + E.ToString + Context.Binding.PeerIP);
        end;
      end;
    end;
  finally
    FreeAndNil(Stream);
    Context.Connection.IOHandler.WriteBufferClose;
    if DeleteFiles(zipFileName) = false then console.Lines.Add(zipfilename + ' konnte nicht gelöscht werden.');
  end;
end;
```

Abbildung 29: Die Prozedur zum Versenden der Dateien

Um eine Datei zu senden, benötigt die Methode *Sendfile* als Argumente den Dateinamen, welcher den kompletten Dateipfad beinhaltet sowie den Context der Verbindung. Nach der positiven routinemäßigen Überprüfung, ob die Datei existiert, wird die Datei in ein ZIP Archiv verpackt. Da die Header von ZIP Dateien sehr klein sind, müssen nicht unbedingt alle Dateien in ein Archiv verpackt und zusammen gesendet, sondern können ohne größere Verluste der Performance einzeln verpackt übertragen werden.

Um die Datei zu versenden, wird für sie eine Instanz eines *TFileStream*s erzeugt, der die Datei einlesen soll. Der Konstruktor eines FileStreams erhält als Parameter den Dateinamen sowie die Information darüber, wie die Datei geöffnet werden soll. Dieser *mode* enthält den Öffnungsmodus und optional den Freigabemodus. Als Öffnungsmodus benutzt unser FileStream *fmOpenRead*, welches vorgibt, dass die Datei ausschließlich zum Lesen geöffnet wird. Als Freigabemodus wird hier *fmShareDenyWrite* benutzt, um die Datei ausschließlich zum Lesen für andere Anwendungen freizugeben. Dies ist wiederum Grundlage dafür, dass die gleiche Datei zur selben Zeit mehrfach an verschiedene Clients gesendet werden kann. Würde kein Freigabemodus eingerichtet werden, wäre die Datei vom Stream blockiert und könnte nicht von außerhalb des Streams gelesen werden.

Ist der Stream eingelesen, kann dessen Größe an den Client gesendet werden. Dann wird ein Puffer mit *WriteBufferOpen* derKlasse *TidBuffer* instanziiert. Dann kann der Stream über die

Write Methode des IOHandlers gesendet werden. Nach erfolgreicher Übersendung des Streams wird der Puffer wieder mit *WriteBufferClose* geschlossen, der Stream freigegeben und die ZIP Datei gelöscht.

4.5.4 Empfang der Dateien

Bevor auf der Clientseite Dateien empfangen werden können, muss zunächst geprüft werden, ob die nötige Ordnerstruktur besteht. Ist das nicht der Fall, werden die entsprechenden Ordner und Unterordner erstellt. Dann wird ein *TFileStream* zum Schreiben instanziiert. Dabei werden analog zum Stream des Servers der Dateiname, Öffnungsmodus und Freigabemodus festgelegt. Hier wird als Öffnungsmodus fmCreate verwendet, da eine Datei erstellt werden soll. Der Freigabemodus ist fmShareExclusive. Dieser verhindert, dass andere Anwendungen die Datei zum Lesen oder Schreiben öffnen können. Ist der Filestream erzeugt, kann er über die Methode *ReadStream* des IOHandlers des TCPClients geschrieben werden. Anhand der Größe des Streams, die vor der Übertragung gesendet wird, erkennt der Stream, wann die Datei fertig geschrieben ist und gibt sie frei. Dann kann der Stream freigegeben werden und die Datei ist übertragen worden.

4.5.5 Lese- und Schreibrechte

In den Kapiteln 4.5.3 und 4.5.4 wurde beschrieben, wie FileStreams zum Transport von Dateien genutzt werden. Dieses Kapitel soll sich nun genauer mit der Problematik der Öffnungs- und Freigabemodi bei FileStreams beschäftigen.

Sehen wir uns dazu zunächst die Öffnungsmodi an. *fmCreate* erstellt eine Datei mit dem angegebenen Namen. Ist die Datei bereits vorhanden, wird diese zum Schreiben geöffnet. *fmOpenRead* öffnet die Datei ausschließlich zum Lesen, während *fmOpenWrite* ausschließlich zum Schreiben öffnet. *fmOpenReadWrite* öffnet die Datei so, dass der aktuelle Inhalt geändert werden kann, ohne dass er ersetzt wird. Für die Serveranwendung ist *fmOpenRead* sicherlich die beste Wahl, da Dateien nur gelesen werden sollen. Für den Client wird *fmCreate* verwendet. Möglicherweise könnte man hier noch *fmOpenReadWrite* nutzen, jedoch sollen die Dateien nicht ausgelesen, sondern nur überschrieben oder erstellt werden.

Die Wahl der Freigabemodi stellt sich als interessanter heraus. Sehen wir uns auch hier die verschiedenen Parameter an. *fmShareCompat* gibt an, dass andere Anwendungen auf die Datei

entsprechend der FCB-Öffnungsmethode zugreifen können. FCB (*File Control Block*) verwaltet, vor allem bei Servern, den Zugriff auf eine Datei, die mehrere Namen besitzt.[112] *fmShareExclusive* gewährt keiner anderen Anwendung den Zugriff auf die Datei. *fmShareDenyWrite* lässt anderen Anwendungen das ausschließliche Leserecht, während *fmShareDenyRead* Rechte ausschließlich zum Schreiben, aber nicht zum Lesen vergibt. *fmShareDenyNone* lässt zu, dass andere Anwendungen die Datei ohne Einschränkungen zum Lesen oder Schreiben öffnen können. *fmShareExclusive* blockiert den Zugang zur Datei.

Die Serveranwendung öffnet Dateien mit *fmShareDenyWrite,*[113] die Clientanwendung mit *fmShareExclusive*.[114] An dieser Stelle können Probleme insbesondere mit ausführbaren Dateien auftreten. Wenn das Updatetool eine Anwendung überschreibt, was bei Delphi Anwendungen häufig vorkommt, da sie kaum externe Ressourcen beziehen, sondern in die EXE-Datei einbinden, könnte es zu Fehlern kommen, da das Updatetool versucht, die Datei zu blockieren. Das ist auch sinnvoll, da das Überschreiben der Anwendung vom Updatetool, während sie geöffnet ist, zu Fehlern führen könnte.

Deshalb muss verhindert werden, dass die Anwendung läuft, während sie aktualisiert wird. In den meisten Fällen wird die Anwendung auch aktiv sein, wenn sie überschrieben werden soll, da das Updatetool aus der Anwendung heraus gestartet werden kann. Dazu wird ein Schnappschuss (engl.: Snapshot) der Prozessliste von Windows gemacht und überprüft, ob Anwendungen, die sich in der Dateiliste des Updates befinden, gestartet wurden.[115] Dazu wird eine Liste von Dateien aus dem temporären Downloadverzeichnis erstellt, die die Endung EXE besitzen, also ausführbar sind. Dann wird nach jedem Dateinamen im Snapshot gesucht. Sind Anwendungen geöffnet, die aktualisiert werden sollen, öffnet sich ein Fenster mit einer Liste der betreffenden Programme. Der Nutzer wird aufgefordert, die Programme zu schließen. Dann werden in regelmäßigen Abständen neue Snapshots der Prozessliste gemacht, um zu überprüfen, ob die Anwendungen beendet wurden. Erst dann fährt das Updatetool seine Arbeit fort. Möchte der Nutzer seine Programme nicht manuell beenden, erhält er die Möglichkeit vom Updatetool, sie schließen zu lassen. Zusätzlich kann er in per Checkbox wählen, ob die beendeten Anwendungen nach dem Update wieder gestartet werden sollen.

Durch diese Maßnahme wird verhindert, dass es zu Schreib- oder Lesekonflikten beim Update kommen kann.

---

[112] Vgl.: http://msdn.microsoft.com/en-us/library/windows/hardware/ff556792(v=vs.85).aspx
[113] Vgl.: Kapitel 4.5.3 Senden der Dateien
[114] Vgl.: Kapitel 4.5.4 Empfang der Dateien
[115] Vgl.: Abbildung 30

```
function TCloseAppsDialogForm.TaskCheck : boolean;
var
  i: integer;
  h: THandle;
  p: TProcessEntry32;
  exists : boolean;
  TempStr : string;
  ProzessID : cardinal;
begin
  Result := False;

  SetLength(ExeList, Length(ExeRunning));
  for i := 0 to Length(ExeRunning) - 1 do
  begin
    ExeList[i] := ExeRunning[i].FileName;
  end;

  SetLength(ExeRunning, 0);
  for i := 0 to Length(ExeList) - 1 do
  begin
    p.dwSize := SizeOf(p);
    h := CreateToolHelp32Snapshot(TH32CS_SnapProcess, 0);
    try
     Process32First(h, p);
     repeat
      TempStr := '';
      ProzessID := 0;
       exists := AnsiUpperCase(ExeList[i]) = AnsiUpperCase(p.szExeFile);
       if exists then
       begin
        Tempstr := (ProcessFileName(p.th32ProcessID));
        ProzessID := p.th32ProcessID;
       end;
     until exists or (not Process32Next(h, p));
    finally
     CloseHandle(h);
    end;

    if exists then begin
      Result := True;
      SetLength(ExeRunning, Length(ExeRunning) + 1);
      ExeRunning[Length(ExeRunning) - 1].FileName := ExeList[i];
      ExeRunning[Length(ExeRunning) - 1].Pfad := TempStr;
      ExeRunning[Length(ExeRunning) - 1].ProzessID := ProzessID;
    end;
  end;
end;
```

Abbildung 30: Die Funktion TaskCheck überprüft, ob Anwendungen gestartet wurden, die in der Dateiliste des Downloads stehen

4.6 Update unter Windows Vista oder höher

Durch das neue Sicherheitskonzept von Windows, das ab der Version Vista greift, ist es schwierig geworden, in Ordner zu schreiben, die keine allgemeinen Schreibzugriffe erhalten. Dafür werden dann Administratorrechte benötigt. Möchte man zum Beispiel in die Programmverzeichnisse der VVW Software schreiben, wird der normale Nutzer ohne Rechte aufgehalten. Entweder muss dann der Administrator einberufen werden oder die Zugriffsrechte der Ordner angepasst werden. Da ersteres bedeutenden Mehraufwand und letzteres Sicherheitsprobleme bedeuten würden, muss ein anderer Weg gewählt werden.

Für Client-Server Versionen ist es sehr wahrscheinlich, dass die Nutzer der Programme über keine Administratorrechte verfügen. Deshalb erhalten sie zusätzlich zur Clientanwendung das Updatetool als Dienst.[116] Dieser erhält bei der Installation alle nötigen Rechte.

Nutzer der Einzelplatzversionen hingegen sollen vorerst nur die Clientanwendung erhalten. Hier wird davon ausgegangen, dass sie in der Regel selbst der Administrator sind oder dieser schnell zu erreichen ist. Wird ein Update installiert, muss sichergestellt werden, dass das Updatetool über Administratorrechte verfügt, um in alle nötigen Programmpfade schreiben zu können. Man kann also ein Windows Manifest[117] in das Tool integrieren, dann würde im Fall des eingeschränkten Windows Benutzers beim Programmstart die Benutzerkontensteuerung (UAC) aufgerufen werden und der Nutzer zur Eingabe des Administratorkontos gebeten werden.

Durch die Anforderung, dass beim Start eines beliebigen Programms der VVW Software nach Updates gesucht werden sollen, ergibt sich, dass bei jedem Programmstart unter eingeschränkten Rechten das Updatetool gestartet wird und nach dem Administrator fragt. Das ist nicht nur für den Kunden nervig, sondern auch für den Administrator, gerade wenn es sich nicht um dieselbe Person handelt.

Deshalb wurden die Suche und der Download von Updates von deren Installation getrennt. So kommt es, dass der Start des Updatetools keine privilegierten Rechte mehr benötigt. Stattdessen wird, wenn Dateien heruntergeladen wurden, eine Anwendung ohne GUI gestartet, die nur dafür zuständig ist, die Dateien zu entpacken und in die Zielverzeichnisse zu kopieren. Dadurch wird der Nutzer nur noch zur Eingabe der Administratordaten gefragt, wenn er Updates installiert.

4.7 Der Weg vom Entwickler zum Kunden

Im alten Updatetool wurden wie bereits beschrieben[118] Dateien vom PC des Entwicklers zum Updateserver über Port 80 in verpackter Form gesendet. Dabei wurde über eine DLL erkannt, dass es sich um Updatedateien und nicht um http-Anforderungen handelt. Das dabei Sicherheitsbedenken auftreten können, wurde im selben Kapitel auch erklärt.

Diese Vorgehensweise wird sich nun grundlegend ändern. Ausgehend davon, dass sich auf dem Webserver Informationen befinden, die ausschließlich für das Updatetool relevant sind, kann

[116] Vgl.: Kapitel 4.4 Das Updatetool als Windows Dienst
[117] Vgl.: Kapitel 2.4.3.1 Windows Manifeste integrieren
[118] Vgl.: Kapitel 2.2.2 Mängel/Nachteile

man sich überlegen, den Webserver, auf dem die Homepage und das Forum laufen, vom Update zu trennen. Dabei ist die einfachste Lösung, die Logik des Updatesystems auf eine virtuelle Maschine des Servers auszulagern und über eine festgelegte Subdomain ansteuern zu lassen.

Auf der virtuellen Maschine können nun Ordner für die Update Dateien angelegt werden, welche vom Entwickler zum Beispiel über einen FTP – Zugang übertragen werden können. Für mehr Sicherheit durch eine verschlüsselte Verbindung kann hier die Nutzung des SFTP Protokolls sorgen. Das **Secure File Transfer Protocol** oder *SSH File Transfer Protocol* (Secure Shell Transfer Protocol) stellt eine sichere Verbindung zwischen dem Updateserver und der Maschine des Entwicklers her. Ein *WinSCP*[119] Client erstellt einen geschützten Tunnel und erschwert Angreiffern durch Benutzerkonten und Verschlüsselung der Verbindung den Zugriff auf den Transfer. Geplant ist eine kleine Anwendung, die *WinSCP* ersetzen soll und automatisch Dateien über das SFTP auf den Server übertragen soll.

Durch die Trennung von Homepage und Update Server auf verschiedene Domains kann der TCPServer über Port 80 lauschen. Würden Homepage und Updateserver auf derselben Domain erreichbar sein, könnte der TCPServer nicht am http-Port lauschen, da dieser vom Webserver blockiert wird. Das hat den Vorteil, dass die Verbindung zwischen TCPClient und TCPServer nicht getunnelt werden muss und keine Ports an der Firewall der Clientseite freigegeben werden müssen.

[119] Vgl.: http://winscp.net/eng/docs/lang:de

5 Schlussbetrachtung und Fazit

5.1 Schlussbetrachtung

Ziel dieser Diplomarbeit war es, eine Anwendung zur automatischen Installation von Updates für Softwareanwendungen zu entwickeln. Dazu war es zuerst notwendig, sich einen Überblick über Paketierung und Installation sowie Softwareverteilung und –management zu verschafften. Dazu wurden Methoden und Verfahren vorgestellt. Hier lag der Schwerpunkt auf Kernkomponenten der Softwareinstallation. Neben den technischen Aspekten von Updates wurden ebenfalls wirtschaftliche Interessen betrachtet und Sicherheitsrisiken sowie deren Bekämpfung behandelt. Bei diesem Kapitel mussten klare Abstriche beim Umfang gemacht werden, da die Sicherheit ein wichtiges Thema für die Softwareverteilung und –instandhaltung ist und hier nur grundlegende Dinge angesprochen werden konnten. Davon wären einige Teilgebiete sicherlich geeignet, eine eigene Diplomarbeit zu füllen. Außerdem mussten einige Themen komplett ausgelassen werden. Ein Thema wäre die Softwareverteilung unter anderen Betriebssystemen wie Linux oder MacOS gewesen. Hierbei sind allgemeine Konzepte weiterhin gültig, aber in Punkten wie der Paketverwaltung unterscheiden sich die Betriebssysteme enorm.

Zur Entwicklung des Updatetools wurde unter Berücksichtigung der Anforderungen des Herstellers und den Wünschen der Kundschaft das vorhandene Tool analysiert und Schwächen aufgezeigt. Es wurden wichtige Komponenten zur Entwicklung und die wesentlichen Programmroutinen erklärt sowie verschiedene Problematiken, wie die der Nutzerrechte in Windows, angesprochen und Lösungswege dargestellt.

5.2 Fazit

Das Thema Softwareverteilung und -aktualisierung ist groß und komplex und wird durch die Praxis geprägt. Dass die automatisierte Softwareverteilung in Zukunft eine größere Rolle spielen wird, ist anzunehmen. Dabei können die Anforderungen in Unternehmen sehr unterschiedlich sein. Auch spielt der Grad der Komplexität der in einem Unternehmen eingesetzten Software eine Rolle. Dadurch ergeben sich auch Chancen für kleinere Projekte. Viele Unternehmen, die ihre Software im Moment noch per Hand verteilen, werden mittelfristig auf entsprechende Softwaremanagementlösungen umsteigen. Dabei kommen die in dieser Arbeit vorgestellten Systeme zum Einsatz, möglicherweise mehrere von ihnen in Kombinationen.

Das Thema Softwareupdate bzw. Aktualisierung spielt heute schon eine große Rolle und wird immer mehr an Bedeutung gewinnen, da Software immer häufiger verändert wird und häufig nicht mehr in Abständen in großen Paketen sondern mit kleineren Veränderungen, dafür aber öfters aktualisiert wird. Dadurch wird die Automatisierung von Softwareupdates immer interessanter. Ein wichtiger Aspekt der Updates ist ihre Sicherheit. Zu oft werden Updates über unsichere Verbindungen eingespielt. Viele Betrüger übermitteln ihre Schädlinge über vermeintliche Softwareaktualisierungen, was der Nutzer oft nicht bemerkt. Hier ist sicherlich der größte Handlungsbedarf bei Softwareherstellern.

Literaturverzeichnis

Bücher

Klaus Chantelau, René Brothuhn: Multimediale Client-Server-Systeme. -1.Aufl. –Schmalkalden: Springer, 2009

Douglas E. Comer: Konzepte, Protokolle, Architekturen. -1.Aufl. –Hemsbach: Hüthig Jehle Rehm, 2011

Andre Exner: Secure Socket Layer (SSL). -1.Aufl. –Norderstedt: GRIN Verlag, 2008

Thomas Joos: Planungsbuch Microsoft-Netzwerke. -1.Aufl. –München: Addison-Wesley Verlag, 2006

Eugene Kaspersky: Malware. -1.Aufl. –Moskau: Hanser Verlag, 2008

Frank A. Koch: Computer-Vertragsrecht. -7.Aufl. –München: Haufe-Lexware, 2009

Dirk Larisch: Netzwerktechnik. -3.Aufl. –Heidelberg: Hüthig Jehle Rehm, 2010

Klaus-Rainer Müller: IT-Sicherheit mit System. -4.Aufl. –Wiesbaden: Springer, 2011

William Panek, Tylor Wentworth: Mastering Microsoft Windows 7 Administration. -1.Aufl. – Indianapolis(USA): John Wiley & Sons, 2010

Klaus Schmeh: Kryptografie : Verfahren, Protokolle, Infrastrukturen. -1.Aufl. - Dpunkt-Verlag, 2009

Thomas Schwenkler: Sicheres Netzwerkmanagement. -1.Aufl. –Soest: Springer, 2005

Holger Schwichtenberg, Thomas Joos, Manuela Reiss: Windows Vista Business: Das Profihandbuch für den Unternehmenseinsatz. -1.Aufl. –München: Addison-Wesley Verlag, 2008

Uwe Schneider, Dieter Werner: Taschenbuch der Informatik. -6.Aufl. Hanser Verlag, 2007

Erich Stein: Taschenbuch Rechnernetze und Internet. -3.Aufl. –Jena: Hanser Verlag, 2008

Bruno Studer: Netzwerkmanagement und Netzwerksicherheit. -1.Aufl. –Zürich(CH): vdf Hochschulverlag AG, 2010

Carlo Westbrook: Active Directory für Windows Server 2008. -1.Aufl. –München: Addison-Wesley Verlag, 2009

Quellen aus dem Internet

Pingdom HQ; support@pingdom.com; Internet 2011 in numbers; <http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/>; 17.01.2012

The Organisation for Economic Co-operation and Development (OECD); Households with access to the Internet; <http://www.oecd.org/dataoecd/>; 2012

The Organisation for Economic Co-operation and Development (OECD); Households with access to a computer at home; <http://www.oecd.org/dataoecd/>; 2012

The Organisation for Economic Co-operation and Development (OECD); OECD fixed broadband penetration net increase; <http://www.oecd.org/dataoecd/>; 2012

The Organisation for Economic Co-operation and Development (OECD); Total number of fixed broadband subscriptions; <http://www.oecd.org/dataoecd/>; 2012

Mareike Wueste; Internet Speeds and Costs Around the World; <http://mareikewueste.files.wordpress.com/2010/01/cwinv.jpg>; 01/2010

Wissen.de; Update; kontakt@wissen.de; <http://www.wissen.de/lexikon/update?keyword=update>; 2012

Dict.cc; paul@dict.cc; up to date; <http://www.dict.cc/?s=up+to+date>; 2012

Wissen.de; Patch; <http://www.wissen.de/fremdwort/patch?keyword=patch>;2012

Microsoft; kunden@microsoft.com; Download Center; <http://www.microsoft.com/en-us/download/default.aspx?>; 2012

uni-protokolle.de; info@uniprotokolle.de; Prozedurale Programmierung; <http://www.uni-protokolle.de/Lexikon/Prozedurale_Programmierung.html>; 2012

Flexera Software; Adminstudio; http://www.flexerasoftware.com/products/adminstudio.htm> ; 2012

Microsoft; kunden@microsoft.com; Office 2010 Resource Kit; < http://technet.microsoft.com/de-de/library/cc303401.aspx>; 2012

Microsoft Office; kunden@microsoft.com; http://office.microsoft.com/de-de/; 2012

Microsoft; kunden@microsoft.com; Custom Actions; http://msdn.microsoft.com/en-us/library/bd8h80ez(v=VS.80).aspx; 2012

Nullsoft; support@winamp.com; Winamp Media Player; http://www.winamp.com/; 2012

SourceForge.net; kichik@users.sourceforge.net; Download NSIS; http://nsis.sourceforge.net/Download; 2012

Eclipse.org; emo@eclipse.org; Eclipse; http://www.eclipse.org/;2012

Microsoft; kunden@microsoft.com; Visual Studio; http://www.microsoft.com/germany/visualstudio/;2012

Microsoft; kunden@microsoft.com; Merge Moduls;http://msdn.microsoft.com/en-us/library/windows/desktop/aa369820(v=vs.85).aspx; 2012

Microsoft; kunden@microsoft.com; Application Virtualization http://technet.microsoft.com/de-de/library/cc843848; 2012

Flexera Software; InstallShield; http://www.flexerasoftware.com/products/installshield.htm; 2012

CASIC; info@casic.de; InstallAware; http://www.installaware.de/; 2012

Jrsoftware.org; Inno Setup; http://www.jrsoftware.org/isinfo.php; 2012

Bitrock; support@bitrock.com; InstallBuilder; http://installbuilder.bitrock.com/; 2012

Microsoft; kunden@microsoft.com; SO WIRD'S GEMACHT: Verwenden von Gruppenrichtlinien für die Remoteinstallation von Software in Windows 2000; http://support.microsoft.com/kb/314934/de; 2012

Microsoft; kunden@microsoft.com; System Center Configuration Manager; http://www.microsoft.com/de-de/server/system-center/2007-2010/configuration-manager/default.aspx; 2012

Microsoft; kunden@microsoft.com; System Center Configuration Manager - Funktionen; http://www.microsoft.com/de-de/server/system-center/2007-2010/configuration-manager/funktionen.aspx; 2012

Opsi; info@uib.de; open pc server integration; http://www.opsi.org/en; 2012

Frontrange; info.fr@frontrange.com; Frontrange; http://www.frontrange.com/; 2012

Msg Systems AG; christian.boellert@msg-services.com; Software Lifecycle Management; http://www.msg-systems.com/sysserv_sw_lifecycle.0.html;2012

Heise.de; netze@heise.de; Frontrange kauft Enteo; http://www.heise.de/netze/meldung/Frontrange-kauft-Enteo-155747.html; 2007

Frontrange; info.fr@frontrange.com; HEAT Client Management; http://www.frontrange.com/client-management/; 2012

Novell; contact-fr@novell.com; Novell; http://www.novell.com/de-de/home/; 2012

Hardware aktulle; Christian.Schade@hardware-aktuell.com; Upgrade; http://www.hardware-aktuell.com/lexikon/Upgrade/;2012

Wikipedia; info-de@wikimedia.org; Adobe Photoshop; http://de.wikipedia.org/wiki/Adobe_Photoshop; 2012

Adobe Sytems GmbH; dublin-web@adobe.com; Adobe Photoshop CS6; http://www.adobe.com/de/products/photoshop.html

Comwave; info@comwave.de; Destinator-7; http://www.comwave.de/GPS---Navigation/Destinator/Destinator-Update-PDA/Destinator-7---Cross-Update---West--und-Osteuropa.html; 2012

Teamviewer GmbH; service@teamviewer.com; TeamViewer, http://www.teamviewer.com/de/; 2012

PCVisit Software AG; kontakt@pcvisit.de; PCVisit Software AG; http://www.pcvisit.de/;2012

Softonic; moc.cinotfos@anolecrab; Softonic Download Manager; http://www.softonic.de/download-manager; 2012

Heise Security; red@heisec.de; Super-Spion Flame trug Microsoft-Signatur; http://www.heise.de/security/meldung/Super-Spion-Flame-trug-Microsoft-Signatur-1590335.html; 2012

Bundesamt für Sicherheit in der Informationstechnik; grundschutz@bsi.bund.de; Informationssicherheit; https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzStandards/ITGrundschutzStandards_node.html; 2012

Vordruckverlag GmbH; info@vordruckverlag.de; VVW GmbH; http://www.vordruckverlag.de/; 2012

Pixelplanet; http://www.xpressupdate.de/kontakt/index.html; XpressUpdate; http://www.xpressupdate.de/; 2012

Pixelplanet; http://www.pixelplanet.de/kontakt/index.html; Pixelplanet; http://www.pixelplanet.de/; 2012

SamLogic Software; support@samlogic.com; What is a Manifest (in Windows) ; http://www.samlogic.net/articles/manifest.htm; 2012

Embarcadero; info@embarcadero.com; DATENBANKWERKZEUGE; http://www.embarcadero.com/de/; 2012

Embarcadero; info@embarcadero.com; Firemonkey; http://www.embarcadero.com/de/products/firemonkey; 2012

Internet Direct; The Indy Project; http://www.indyproject.org/index.de.aspx, 2012

Microsoft; kunden@microsoft.com; The FCB Structure; http://msdn.microsoft.com/en-us/library/windows/hardware/ff556792(v=vs.85).aspx; 2012

OpenLogic; WinSCP; http://winscp.net/eng/docs/lang:de; 2012

Firmenschriften

Flexera Software; InstallShield Help Library

Zeitschriftenartikel

Schmitz, Ludgar: Windows-Client-Management mit dem neuen Opsi 4.0.1. In: Admin Magazin. –München: Medialinx AG. -06/2010

Abbildungsverzeichnis

Erklärung zur selbstständigen Anfertigung der Arbeit

Ich erkläre, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Dresden, 29.10.2012