

Diplomarbeit

Konzeption und Realisierung
einer Erweiterung des PotatoSystems zur
elektronischen Software-Distribution

von
Martin Fürstenau



Universität Potsdam
Institut für Informatik
Professur Netzwerktechnologien und multimediale Teledienste

Aufgabenstellung und Betreuung:
Prof. Dr. Klaus Rebenburg

Potsdam
13. Juni 2006

Abstract

Abstract

Die Verteilung virtueller Güter über das Internet ist seit Mitte der 1990er Jahre eine alternative Vertriebsmöglichkeit für kleine und große Unternehmen. Das community-basierte Potato-System bietet solche eine Plattform an, wobei der Schwerpunkt auf der elektronischen Musik-Distribution liegt. Mit dieser Arbeit wird ein neues Teilsystem eingeführt, das einen weitestgehend automatisierten Vertrieb des Medientyps Software ermöglicht. Ausgehend von einer Beschreibung des aktuellen Systems, werden diverse Konzepte basierend auf den Vorgaben der Systembetreiber entwickelt und auf ihre Zweckmäßigkeit und Machbarkeit hin untersucht. Die Realisierung des verwendeten Konzepts wird beschrieben und zusätzlich auf ihre Schwächen untersucht. Darauf aufbauend werden Verbesserungen für weitere Entwicklungsschritte vorgeschlagen.

Inhaltsverzeichnis

Inhaltsverzeichnis

1	Einleitung	10
1.1	Motivation	10
1.2	Ziel der Arbeit.....	11
1.3	Aufbau der Arbeit	11
2	Elektronische Software Distribution.....	13
2.1	Arten der Software Distribution	13
2.1.1	Begriffliche Abgrenzung	15
2.1.2	ESD als neuer Vertriebsweg.....	16
2.2	Arten von Software	18
2.2.1	Standard- und Individualsoftware	18
2.2.2	Public Domain, Freie Software und Open Source	19
2.2.3	Freeware	20
2.2.4	Shareware und Varianten.....	20
2.2.5	Weitere Formen.....	22
2.3	Software-Lizenzierung	22
2.3.1	Free Software und Open Source Software License.....	23
2.3.2	Non-Free / Non-Open Source License	25
3	Softwarepiraterie	27
3.1	Schaden durch Piraterie.....	27
3.2	Kategorisierung von Piraterie	29
3.3	Schutz vor Piraterie.....	30
3.3.1	Technische Möglichkeiten.....	30
3.3.1.1	Verschlüsselung –Herzstück bestehender Kopierschutzverfahren	31
3.3.1.2	Erweiterte Schutzkonzepte.....	32
3.3.1.3	Serial-Number	34
3.3.1.4	Produktaktivierung	35
3.3.1.5	Software als Service	35
3.3.1.6	Hardware-basierter Schutz.....	36
3.3.1.7	Trusted Computing.....	37
3.3.2	Digital Honesty	39
3.4	Fair Play auf beiden Seiten.....	41
4	Das PotatoSystem	43
4.1	Motivation	43
4.2	Geschäftsmodell und Dienstleistungen	43
4.3	Architektur.....	45
4.4	Abläufe	48
5	Konzeption einer Erweiterung zur elektronischen Software-Distribution.....	51
5.1	Anforderungen an die ESD-Erweiterung	51
5.2	Unterschiede zur Musik-Distribution.....	51
5.3	Kategorisierung der Verteilungsstrategien.....	53
5.3.1	Auslieferung der Software als „Try-and-Buy“-Version.....	53
5.3.2	Auslieferung der Software als „Try-and-Die“-Version.....	58
5.4	Erweiterungen des PotatoSystems	60
5.4.1	Auslieferung.....	60
5.4.2	Verwaltung.....	62
5.4.2.1	Verwendung einer Notfallliste	62
5.4.2.2	Verwendung von Auslieferungslisten	63
5.4.3	Speicherformat der Produktschlüssellisten.....	63

5.4.4	Identifizierung der Produktschlüssellisten.....	65
5.4.5	Persistente Speicherung verwendeter Produktschlüssel.....	65
5.4.6	Produktregistrierung und -Updates	66
5.5	Anforderungen und Erweiterungen auf Anbieterseite	67
5.5.1	Bereitstellung der Produktschlüssellisten	67
5.5.1.1	Identifizierung	67
5.5.1.2	Einheitlicher Zugriff.....	68
5.5.2	Produkt-Updates.....	69
5.6	Erstellen eines Sharewareprogramms als Dienstleistung des PotatoSystems	69
5.7	Sicherheitskritische Aspekte.....	73
5.7.1	Übermittlung der Produktschlüssel und Produktschlüssellisten	73
5.7.2	Produktschlüssel und Einhaltung der Lizenzen	76
6	Realisierung der Erweiterung.....	78
6.1	Aktuelle Konfiguration des PotatoSystems.....	78
6.2	Prototypische Realisierung.....	79
6.2.1	Das ESD-Modul im PotatoSystem.....	79
6.2.2	Das ESD-Skript im geschützten Anbieterverzeichnis.....	84
6.3	Von der Registrierung bis zum Verkauf	85
7	Schwächen der Realisierung und des verwendeten Konzepts	86
8	Zusammenfassung	91
9	Ausblick	93
Anhang A.....		95
Literaturverzeichnis		99

Abbildungsverzeichnis

Abbildungsverzeichnis

Abbildung 1 Methode zur Berechnung der Piraterierate nach IDC	28
Abbildung 2 Provisionsmodell des PotatoSystems	44
Abbildung 3 Einbettung des PotatoSystems in seine Umgebung	46
Abbildung 4 Architektur des PotatoSystems	47
Abbildung 5 einfacher Kaufvorgang	50
Abbildung 6 Kauf eines Produktschlüssels	54
Abbildung 7 ESD-Erweiterung im PotatoSystem.....	61
Abbildung 8 Shareware-Umwandlung als PS-Dienstleistung	72
Abbildung 9 Verwendete Technologien und Systeme des PotatoSystems	78
Abbildung 10 UML Klassendiagramm der realisierten ESD-Erweiterung	80

Abkürzungsverzeichnis

Abkürzungsverzeichnis

(R)DBMS	(Relational) Database Management System
4FO AG	4 FriendsOnly Internet Technologies AG
AES	Advanced Encryption Standard
AS	Accounting Server
ASCII	American Standard Code for Information Interchange
ASP	Application Service Provider
Axis	Apache Extensible Interaction System
BSA	Business Software Alliance
BSD	Berkley Software Distribution
CD	Compact Disk
CPU	Central Processing Unit
CRM	Customer Relationship Management
CSV	Character Separated Value
DES	Data Encryption Standard
DRM	Digital Rights Management
DVD	Digital Versatile Disk
EMA	Enterprise Management Association
EPM	Enterprise Project Management
ESD	Elektronische Software Distribution
FK	Fremdschlüssel (foreign key)
FMC	Fundamental Modeling Concepts
GEMA	Gesellschaft für musikalische Aufführungs- und mechanische Vervielfältigungsrechte
GnuPG	GNU Privacy Guard
GPL	GNU General Public License
GUI	Graphical User Interface
HTTP(S)	(Secure) Hypertext Transfer Protocol
IDC	International Data Coporation
IDMT	Fraunhofer Institut Digitale Medientechnologie
IP	Internet Protocol
JDBC	Java Database Connectivity
JRE	Java Runtime Environment
JSP	Java Server Pages
KISS	Keep it stupid and simple
LAN	Local Area Network
LGPL	GNU Lesser General Public License
LT	LaGrande Technology
MVC	Model-View-Controler
NGSCB	Next-Generation Secure Computing Base
PC	Personal Computer
PGP	Pretty Good Privacy
PHP	PHP Hypertext Processor
PS	PotatoSystem
RFC	Request for Comments
RISC	Reduced Instruction Set Computing
RSA	Rivest, Shamir, Adleman

Abkürzungsverzeichnis

RTM	Roots-of-trust for Measurement
RTR	Roots-of-trust of reporting
RTS	Roots-of-trust of storage
SAVE	Shareware Autoren Vereinigung
SCSL	Sun Community Source License
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Socket Layer
TAN	Transaktionsnummer
TC	Trusted Computing
TCG	Trusted Computing Group
TCPA	Trusted Computing Platform Alliance
TLS	Transport Layer Security
TP	Trusted Platform
TPM	Trusted Platform Module
TSS	TCG Software Stack
UML	Uniform Modeling Language
XML	eXtensible Markup Language

1 Einleitung

1.1 Motivation

Die Softwareindustrie ist durch die technischen Fortschritte auf den Gebieten der Computerhardware und -Software in der Lage, ihre Waren auf verschiedenen Vertriebskanälen zu verbreiten. Die Verteilung über physische Medien wie CD, DVD oder Diskette sind zwar immer noch vorherrschend, allerdings ist eine Verlagerung auf eine nicht-physische Form über bestehende technische Kommunikationskanäle zu beobachten. Dies wird durch die zunehmende Zahl an Breitbandanschlüssen begünstigt. Kommunikationsverbindungen mit hohem Durchsatz waren früher hauptsächlich nur zwischen großen Firmen vorhanden. Jetzt sind auch kleine und mittlere Unternehmen in die Lage versetzt, große Softwarepakete anzubieten und von anderen Anbietern herunter zu laden. Diese Möglichkeit wird auch Privatpersonen zu Teil. Ende 2005 verfügte laut BITKOM jeder vierte deutsche Haushalt über einen Breitband-Internetanschluss (26).

Die Möglichkeit der elektronischen Verteilung kann auf viele Produkte angewendet werden, z.B. eBooks, Musik, Videos oder Software. Diese so genannten virtuellen Waren unterliegen im Allgemeinen einer bestimmten Form von Copyright. Deshalb müssen sie vor unrechtmäßigen Zugriffen und unrechtmäßiger bzw. illegaler Verteilung geschützt werden. Dies wird bei Multimediadateien durch spezielle Programme erledigt, die bspw. zur Wiedergabe der Daten benutzt werden. Die verwendeten Sicherheitskonzepte und Technologien werden unter dem Begriff *Digital Rights Management* (DRM) zusammengefasst. Ein Beispiel ist *Microsofts Windows Media Rights Manager*, integriert im *Microsoft Windows Media Player*.

Softwarehersteller kennen die Problematik illegaler Verbreitung ihrer Produkte seit geraumer Zeit. Relativ neu sind allerdings die neue Qualität und das Ausmaß des Grades illegaler Verbreitung. Softwareprodukte haben die Möglichkeit, sich gegen Piraterieangriffe zu wehren, jedoch ist auf Softwareebene niemals ein 100-prozentiger Schutz möglich. Vielmehr geht es darum, einen Schutzmechanismus zu kreieren, der so lange wie möglich sämtlichen Angriffen und Umgehungsmethoden standhält. Laut der *Business Software Alliance* (BSA) entstand im Jahr 2005 durch Softwarepiraterie ein weltweiter Schaden von ca. 34 Milliarden US Dollar; Tendenz steigend (33).

Die Konsequenzen sind vielfältig und wirken sich auf verschiedenen Ebenen teilweise dramatisch aus. So werden durch die Industrie Schutzmechanismen sowohl auf technischer als auch juristischer Ebene konzipiert, die auf sensible Daten der Nutzer zurückgreifen. Evtl. wird gegen verfassungsmäßige Rechte wie der Schutz der Privatsphäre verstoßen.

Einleitung

Diese Problematik wird seit einiger Zeit durch die Datenschutzbeauftragte der Länder genauer untersucht. Daraus ergibt sich für den Konsumenten manchmal der Eindruck, er sei ein Krimineller. Dies steigert die Popularität von Tauschbörsen wie *Kazaa* oder *emule*.

Damit die unterschiedlichen Belange der Anbieter und Konsumenten nicht nur einseitig beachtet werden, wurde das PotatoSystem (PS) erschaffen. Es ist eine Gemeinschaftsentwicklung der *4FriendsOnly.com Internet Technologies Aktiengesellschaft* (4FO AG (2)), einem Spin-Off¹ der TU-Ilmenau, und dem *Fraunhofer Institut Digitale Medientechnologie* (IDMT). Ziel ist eine faire Behandlung der Interessen aller Parteien. Es wurde ein Geschäftsmodell entwickelt, das diesem Ziel dienen soll. Das System wird zurzeit hauptsächlich für den Musikvertrieb genutzt.

1.2 Ziel der Arbeit

Diese Diplomarbeit hat das Ziel, das PotatoSystem um die Möglichkeit zur Verteilung des Medientyps Software zu erweitern. Diese Art des Vertriebs wird als „Elektronische Software Distribution (ESD)“ bezeichnet. Das System und seine Zielsetzung werden beschrieben und welche Maßnahmen ergriffen werden müssen, um die Erweiterung zu realisieren. Abschließend wird die Realisierung auf Schwächen untersucht und Vorschläge zur Verbesserung in weiteren Entwicklungsstufen unterbreitet.

1.3 Aufbau der Arbeit

Nach einer allgemeinen Einleitung und grundsätzlichen Beschreibung der Hintergründe dieser Arbeit (Kapitel 0), wird die Verteilung von Software sowohl historisch als auch mittels neuer Entwicklungen in elektronischer Form über technische Kommunikationskanäle betrachtet. Zunächst wird eine einheitliche Terminologie herausgestellt, um im weiteren Verlauf den innovativen Gehalt und die Möglichkeiten der als Elektronische Software-Distribution (ESD) bezeichneten Verteilungsform untersuchen zu können (Kapitel 2). Es wird zudem auf die verschiedenen Softwarearten und ihren Lizenzierungsmodellen eingegangen. In diesem Zusammenhang gilt das Augenmerk in Kapitel 3 dem Problem der Softwarepiraterie, ihren Auswirkungen und Möglichkeiten dem entgegenzuwirken.

Basierend auf der Beschreibung des PotatoSystems (Kapitel 4), werden die Grundlagen und Anforderungen der Erweiterung des Systems zur Elektronischen Software-Distribution aufgezeigt (Kapitel 5)

¹ Spin-Off bezeichnet hier die Neugründung eines Unternehmens aus einem universitären Projekt heraus

Einleitung

Die herausgearbeiteten Konzepte werden bzgl. ihrer Zweckmäßigkeit, Machbarkeit und den gestellten Anforderungen untersucht. Die Realisierung (Kapitel 6) wird mittels Problemanalyse bewertet und Verbesserungen vorgeschlagen (Kapitel 7).

Abschließend werden die wesentlichen Punkte zusammengefasst und ein Ausblick auf zukünftige Entwicklungsmöglichkeiten gegeben (Kapitel 8).

2 Elektronische Software Distribution

Der Vertrieb und die Verteilung von Softwareprodukten erfahren mit der steigenden Funktionalität von informationsverarbeitenden Systemen einen Wandel. Immer mehr Unternehmen bieten ihre Waren im Internet zum Download an. Im Folgenden werden die Verteilung über herkömmliche physische Medien und die rein virtuelle Form betrachtet. Zusätzlich werden verschiedene Softwarearten und ihre Lizenzierungen beschrieben, um zu verdeutlichen, dass ESD nicht für alle Programme gleichermaßen zweckmäßig ist

2.1 Arten der Software Distribution

Die ersten elektronischen Rechenmaschinen des 20. Jahrhunderts, teilweise riesige Maschinen², führten Programme aus, die mittels gelochten Filmstreifen codiert wurden. Durch technischen Fortschritt auf den Gebieten der Hardware gelangten die Programme und ihre Daten auf Magnetbänder und –Scheiben, bspw. die Datasette bzw. die Diskette. Dank des wahlfreien Zugriffs auf die gespeicherten digitalen Daten einer Diskette wurde sie häufiger zum Transport eines Programms von einem Rechner auf einen anderen genutzt.

Mit der Zeit wurden Programme immer komplexer und ihre Größe nahm zu. Die Kapazitäten von Disketten genügten nicht mehr den Ansprüchen von Unternehmen und Konsumenten. So wurden komfortablere, stabilere und zur konsistenten Datenhaltung besser geeignete Datenträger entwickelt. Dies sind Datenträger wie die *Compact Disk* (CD) und deren Nachfolger, die *Digital Versatile Disk* (DVD), die auf optischen Speichereigenschaften beruhen. An Daten kann eine CD bis zu 900 MByte, eine DVD bis zu 17 GByte fassen (56). Trotzdem werden noch heutzutage zur Datensicherung und Archivierung großer Datenbestände Magnetbänder genutzt. Bei ihnen handelt es sich um ein lineares Speichermedium, das im Gegensatz zur CD oder DVD nur sehr unzureichend wahlweisen Zugriff auf die gespeicherten Daten zulässt.

Das Medium CD ist immer noch das meist genutzte Vertriebsmittel für Software- als auch für Musikunternehmen. Eine Ausnahme bildet hier die Filmindustrie, deren Werke hauptsächlich auf DVD zu erhalten sind. Dies wird ebenfalls beim Vertrieb von Softwareprodukten über physische Medien immer häufiger der Fall werden. Schon heute ist zu beobachten, dass Konsumenten bei umfangreichen Produktsammlungen die Möglichkeit erhalten, zwischen einer großen CD-Kollektion oder einer einzelnen bzw. wenigen DVD(s) zu wählen. Händlern erlaubt die Kapazität einer DVD zum eigentlichen Produkt evtl. weitere Zusatzprogramme auf demselben Datenträger anzubieten.

² Eine der größten Maschinen war die in Amerika 1945 entwickelte „MARK I“ (57). Sie war über 15 Meter lang und ca. 2,5 Meter hoch

Elektronische Software Distribution

Die Verteilung über physische Medien ist trotz der Möglichkeit des Outsourcings mit hohen Reproduktions- und Vertriebskosten verbunden³. Die Auslieferung der entwickelten Produkte setzt die Existenz sämtlicher Hardwarekomponenten voraus. Bspw. müssen die zum Brennen verwendeten Rohlinge eingekauft, gelagert und passende Brenner bereitgestellt werden. Weitere Kosten entstehen durch die Auslieferung an Händler, welche die Produkte zum Verkauf anbieten. Im Normalfall wird dies nicht durch ein einzelnes Unternehmen realisiert. Vielmehr haben verschiedene Parteien Anteil an der Verteilung. Sie alle spielen verschiedenen Rollen, die wie folgt zusammengefasst werden können⁴:

- der Softwarehersteller – entwickelt Produkte und bereitet sie zur Auslieferung vor
- der Verteiler – erhält vom Softwarehersteller den Auftrag, die Produkte auszuliefern
- der Wiederverkäufer – verkauft die erhaltenen Produkte vornehmlich an Einzelhändler
- der Einzelhändler – verkauft die erhaltenen Produkte an Endnutzer.
- der Endnutzer – erwirbt Produkte vom Hersteller, Wiederverkäufer oder Einzelhändler

Einzelne Unternehmen können mehrere Rollen ausfüllen. Z.B. kann ein Softwarehersteller gleichzeitig Verteiler und/oder Wiederverkäufer sein.

Die Risiken einer physischen Distribution sind zahlreich. Grob können sie in finanzielle, technische und Reputations-Risiken unterteilt werden. Sie stehen in Wechselbeziehungen zueinander.

Aus finanzieller Sicht betrachtet, haben Softwarehersteller das grundsätzliche Problem des Warenwerts eines Produkts. Software ist an sich ein immaterielles Gut. Die Kosten lassen sich durch die Entwicklungsinvestitionen errechnen, z.B. Anzahl und Kosten der Mitarbeiter, Wert der genutzten Entwicklungs- und Testsysteme, etc. Durch die Bindung an ein physisches Medium entstehen weitere Kosten für Einkauf und Lagerung der Rohlinge und Kopien, der Reproduktion und der Auslieferung. Speziell bei der Auslieferung können Logistikprobleme auftreten, z.B. Verspätungen, Unfälle oder Raub beim Transport. Außerdem stellt sich die Frage, wie viele Exemplare eines Produkts bereitgestellt werden sollen. Die Herstellung zu vieler Kopien kostet ein Unternehmen wertvolles Kapital, bei zu wenigen entgehen mögliche Gewinne.

³ Outsourcing bedeutet hier, dass die Produktion in Ländern durchgeführt wird, in denen die Kosten für Maschinen und Arbeiter teilweise deutlich geringer sind als in Industrienationen. Die Arbeit wird entweder von der Firma selbst durchgeführt oder durch einen externen Dienstleister

⁴ Diese Einteilung ist nur grob und keineswegs vollständig. Für den hier betrachteten Kontext ist dieser Ausschnitt ausreichend.

Elektronische Software Distribution

Technische Risiken entstehen hauptsächlich bei der Reproduktion und beim Endnutzer. Die Reproduktion eines Produkts kann aufgrund eines Hardwarefehlers oder Fehlverhaltens seitens der Mitarbeiter ausfallen oder sich verzögern. Der Endnutzer hat i. A. durch den Erwerb eines Produkts die Möglichkeit, dies durch technische Hilfsmittel zu vervielfältigen und durch Umgehung des Kopierschutzes und/oder der evtl. Registrierungsmethoden illegal anderen Konsumenten zugänglich zu machen. Das Problem der Softwarepiraterie wird in Kapitel 3 eingehender erläutert.

Die Reputation eines Softwareherstellers hängt davon ab, in wie weit vertragliche Leistungen eingehalten werden. Das betrifft insbesondere die Funktionalität und Handhabung der Produkte, aber auch deren fristgerechte Auslieferungen. Zusätzlichen Leistungen den Kunden gegenüber werden meist mit erhöhter kundentreue honoriert. Bezüglich der physischen Verteilung von Softwareprodukten hängt die Reputation von den zusätzlichen Lieferungen und Leistungen des Herstellers ab: ist ein gutes, verständliches Handbuch beigelegt, wie zufrieden stellend funktioniert der Support bei verschiedenen Problemen wie bspw. Installationsschwierigkeiten aufgrund unlesbarer Datenträger, ist ein Umtausch möglich, etc.

Es ist offensichtlich, dass die Wechselwirkungen dieser Kategorien starken Einfluss aufeinander haben. Sollte bspw. die Auslieferung eines Produkts aufgrund technischer Schwierigkeiten verzögert werden, so hat dies zum einen finanzielle Konsequenzen, zum anderen wird der Ruf des betroffenen Unternehmens Schaden nehmen.

Die Alternative zur physischen Verteilung stellt die elektronische Softwareverteilung (ESD – *electronic software distribution*) dar. Der Begriff ESD ist nicht eindeutig definiert. Deshalb wird nachfolgend ein einheitlicher Terminus herausgestellt, der auf vorhandenen Beschreibungen und deren Erläuterungen basiert.

2.1.1 Begriffliche Abgrenzung

Unter dem Begriff der elektronischen Softwareverteilung wird zum einen das Anbieten von Softwareprodukten und Lizenzen zum Kauf und Download über bestehende technische Kommunikationskanäle verstanden. Die erworbene Software kann sofort installiert und genutzt werden.

Zum anderen wird ESD als ein Software-Management-System verstanden, das in der Lage ist, Software innerhalb eines lokalen Netzwerks (LAN – *local area network*) automatisch zu verteilen, zu installieren und nach vorhandenen oder gewünschten Vorgaben zu konfigurieren. Dabei handelt es sich hauptsächlich um Firmennetzwerke, die auf einer zentralen Client/Server-Architektur basieren. Der gesamte Vorgang wird von einem „Verteilungsserver“ gesteuert.

Elektronische Software Distribution

Diese gewünschte Vereinfachung soll Netzwerkadministratoren Zeit und Aufwand bei Updates und Einspielen neuer Softwareprodukte sparen. Dies hängt sowohl von der Anzahl der im Netzwerk verwendeten Clients ab, als auch von der Qualität der genutzten Verteilungs-Software.

Viele Unternehmen und ebenfalls Onlinelexika haben diese beiden Sichtweisen kombiniert. ESD ist somit die Möglichkeit, über ein System Software und Lizenzen mittels vorhandener technischer Kommunikationswege zum Kauf und Download anzubieten, zu installieren und evtl. zu aktivieren und/oder registrieren. I.A. haben Kunden die Möglichkeit, die Produkte erst einmal zu testen. Dabei sollen die zentralen Fragen der Zweckmäßigkeit und ob den geforderten Ansprüchen genüge getan wird weitestgehend beantwortet werden⁵. Die Bezahlung kann entweder bei Erhalt der Ware oder nach Ablauf der Testphase erfolgen. Zusätzlich wird jedem Käufer Support angeboten.

Diese Arbeit versteht ESD wie im letzten Fall beschrieben. Es handelt sich also um eine Form des eCommerce.

2.1.2 ESD als neuer Vertriebsweg

Der Vertrieb von Software mittels ESD hat für Softwareunternehmen diverse Vorteile:

- Die entwickelten Produkte können ohne Verzögerung angeboten werden. Sie müssen nicht auf ein physisches Medium gebrannt und dieses anschließend ausgeliefert werden. Sie behalten vollständig ihren virtuellen Charakter. Daher können mittels ESD angebotene Waren als virtuelle Waren bezeichnet werden. Software realisiert Virtualität mittels digitaler Daten, weswegen auch der Begriff „Digitale Güter“ verwendet wird. Nach (9) haben digitale Güter gegenüber realen Waren drei sie unterscheidende Merkmale:
 - Veränderbarkeit – digitale Daten lassen sich leicht modifizieren
 - Reproduzierbarkeit – Kopien sind in kurzer Zeit verfügbar, mit geringem Kostenaufwand und vom Original nicht zu unterscheiden
 - Unzerstörbarkeit – digitale Güter erleiden keine Verschleißerscheinungen
- Digitale Güter sind aufgrund ihrer Virtualität theoretisch jederzeit an jedem Ort verfügbar. Somit sind die Möglichkeiten einer Unternehmens-Expansion erhöht. Durch Web-Auftritte können Kunden angesprochen werden, die durch eine physische Verteilung nicht zu erreichen gewesen wären.

⁵ Dies entspricht dem heutigen Verständnis von Shareware (siehe Kapitel 2.2.4 Shareware und Varianten)

Elektronische Software Distribution

- Der Wegfall der physischen Träger wie CD oder DVD spart einem Unternehmen viele Kosten. Die Probleme der Logistik sind vollständig entfernt. Hardwareprobleme wie das Einkaufen der Rohlinge und Brenner, Reproduzieren und Ausliefern der Kopien entfallen. Der Support kann vereinfacht und effizienter gestaltet werden.
- Die Beziehungen zu den Kunden können besser administriert werden. Das setzt allerdings den Einsatz von Registrierungen voraus. Ein Unternehmen kann so das Verhalten seiner Kunden besser überschauen und verstehen. Die Wahrscheinlichkeit, einen Dialog aufzubauen und somit neue Geschäftsfelder oder Verkaufsmöglichkeiten zu erschließen, ist erhöht.

Es stellt sich für ein Unternehmen also weniger die Frage, ob ESD sinnvoll wäre. Vielmehr ist die Art und Weise interessant. Die Übermittlung von Software von einem Punkt A zu einem anderen Punkt B ist viel komplexer und herausfordernder, als es auf den ersten Blick scheint. Es müssen sowohl technische als auch betriebswirtschaftliche Fragen beantwortet werden.

Im technischen Bereich hat die Sicherheit bei der Auslieferung via Download oberste Priorität. Zum einen muss der Zugriff auf ein Produkt mit dem Ziel der Übertragung einer Kopie, als auch die Übertragung selbst sicher sein. Dazu müssen verschiedenste Sicherheitsmechanismen und -Protokolle verwendet werden. Zum anderen gilt es, das Produkt selbst zu schützen (siehe Kapitel 3 Softwarepiraterie). Daran knüpft sich die Frage der Verfügbarkeit. Ein über ESD angebotenes Softwareprodukt sollte jederzeit zum Download verfügbar sein, um die Möglichkeit einer hohen Verbreitung zu gewährleisten und verschiedenste Konsumentenfragen befriedigen zu können. Zusätzlich kann die Bandbreite bzw. der Durchsatz eine Rolle spielen. Das ist von der Größe eines Produktes als auch von der Anzahl von Downloadanfragen abhängig. Allgemein sollte ein Softwareanbieter oder ein beauftragter Verteiler die Kapazitäten besitzen, große Produkte zum Download mit hohen Durchsatzraten anbieten zu können. Dies sind nur einige Aspekte, die es bei ESD zu bedenken gilt.

Aus wirtschaftlicher Sicht sind vor allem Managementfragen zu beantworten. Die Rückführung der Investitionen, um weiterhin wettbewerbsfähig zu sein, steht dabei an vorderster Stelle. Dies kann nur durch zufriedene Käufer ermöglicht werden, die die Waren erwerben, nutzen und evtl. weiterempfehlen. Ziel muss der Aufbau einer großen Stammkundschaft sein. Es gilt, die vorhandenen Kundenbeziehungen zu analysieren, die wertvollsten zu identifizieren und zu managen. Diese Strategie wird als *Customer Relationship Management (CRM)* bezeichnet. Die internen Prozesse müssen ebenfalls diesem Ziel angepasst werden.

Elektronische Software Distribution

Beschließt ein Unternehmen, die vorhandenen und zukünftig hergestellten Produkte über ESD zu vertreiben, stellt sich die Frage, ob und wie das ESD-Konzept in die vorhandenen IT Geschäftsprozesse integriert werden kann. Laut einer Umfrage, durchgeführt von Enterprise Management Associates (EMA), ist dies die größte Herausforderung.⁶ Ist die Einbettung eines vollständigen ESD möglich, muss ein Unternehmen entscheiden, ob ein eigenes Verteilungssystem entwickelt werden soll. Der Aufwand kann allerdings so enorm sein, dass eine Eigenentwicklung als nicht zweckmäßig erscheint. Einerseits muss der neue Service bekannt gemacht werden, andererseits müssen die Mitarbeiter und Kunden geschult und mit der neuen Art der Auslieferung vertraut gemacht werden. Damit die Organisationsstrukturen und internen Abläufe nicht zu stark modifiziert werden müssen, nutzen viele Softwarehersteller daher die Möglichkeit, die Verteilung durch spezialisierte Drittanbieter durchführen zu lassen. Das bedeutet ein Hinaustragen eigener Geschäftsbereiche und –Abläufe, was gemeinhin unter dem Begriff *Outsourcing* verstanden wird (27).

2.2 Arten von Software

So wie es unterschiedliche Vertriebsarten von Softwareprodukten gibt, so muss der Begriff Software selbst definiert werden. Software wird weithin als alle nicht-physikalische Bestandteile eines Computersystems definiert. Dieser Sammelbegriff ist sehr ungenau und wird im Folgenden weiter differenziert werden. Ausgehend von einer groben Einteilung, werden verschiedene Softwareformen bzgl. ihrer technischen und funktionellen Eigenschaften betrachtet. Die Art der Lizenzierung einer Softwareform wird in Kapitel 2.3 Software-Lizenzierung genauer betrachtet.

2.2.1 Standard- und Individualsoftware

Software kann grob in Standard- und Individualsoftware kategorisiert werden. Standardsoftware ist „[...]auf Allgemeingültigkeit und mehrfache Nutzung hin ausgelegt. In Abgrenzung zur Individualsoftware wird Standardsoftware von Softwarefirmen für einen anonymen Abnehmer entwickelt. Merkmale von Standardsoftware sind somit Anwender- und Branchenneutralität sowie eine auf ein bestimmtes Einsatzgebiet abgestimmte Funktionalität. [...]“.(24) Standardsoftware kann in Applikationssoftware, Systemsoftware und Softwareentwicklungstools differenziert werden.

⁶ Nach IDC ist dies der Grund, warum ESD nur zu 1% im aktuellen Softwaremarkt vorhanden ist. Bezieht man sich nur auf den Auslieferungsteil bzgl. ESD, so haben schon viele Unternehmen die Basis für entsprechende Systeme gelegt

Elektronische Software Distribution

Unter Applikationssoftware versteht man alle Programme, die es Nutzern ermöglicht, eine bestimmte Tätigkeit oder Funktion auszuführen. Das Einsatzgebiet wird nach Business- und Privatbereich unterteilt. Businesssoftware kann für einen speziellen Bereich entwickelt worden sein (*vertical industry applications*) oder sektorübergreifend. Unter letztere fallen Applikationen, die auch im Privatbereich genutzt werden können, z.B. Textverarbeitung, Tabellenkalkulationen, E-Mailprogramme, etc.

Systemsoftware steuert alle internen Abläufe eines Computers. „[...]Sie fungiert als Schnittstelle zwischen Applikationssoftware und den Hardwarekomponenten eines Computers. Systemsoftware kann unterteilt werden in Betriebssysteme, Middleware, System-Management und Security-Lösungen. [...]“(24). Sie stellen die Basis für Applikationssoftware.

Softwareentwicklungstools umfassen alle Programme zum Schreiben und Verwalten von Softwarequellcodes. Dazu zählen Entwicklungs- und Laufzeitumgebungen (z.B. Eclipse, Java Runtime Environment (JRE)), als auch Programme zur Unterstützung des Entwicklungsprozesses (z.B. Microsoft Enterprise Project Management (EPM)).

Das Komplement zur Standardsoftware ist die für einen bestimmten Anwendungsfall entwickelte Individualsoftware. Diese kundenspezifischen Programme sind an die gegebenen Anforderungen der Anwender angepasst. Solche maßgeschneiderten Lösungen sind zum einen in dem jeweiligen Einsatzgebiet sehr flexibel. Zum anderen ist die Entwicklung mit einem hohen Zeit- und Kostenaufwand verbunden. Somit wird Individualsoftware fast ausschließlich proprietäre Programme umfassen.

Standardsoftware ist aufgrund der Allgemeinheit meist kostengünstiger als Individualsoftware. Sie ist in verschiedenen Varianten verfügbar, die nachfolgend in vier weitere Kategorien eingeteilt werden können.

2.2.2 Public Domain, Freie Software und Open Source

Die Softwarevarianten Public Domain, Freie Software und Open Source stellen den Freiheitsgedanken an oberste Stelle. Eine solche Software ist frei verfügbar und darf ohne Einschränkung genutzt werden. Im Falle von Public Domain können die Quellcodes komplett frei erhältlich sein, müssen es allerdings nicht. Falls sie es sind, darf jeder aus ihnen lernen, sie kopieren und modifizieren, sowie die modifizierten Versionen und Varianten weitergeben. Im Falle von Freie Software und Open Source müssen die Quellcodes frei zugänglich sein. Die entwickelten Programme sind durch die an einem Projekt interessierten Beteiligten erschaffen. Der oder die Urheber verzichten im Falle einer Public Domain Software völlig auf ihre Urheberrechte und erwarten keine Vergütung. Public Domain Software ist nicht automatisch mit Freie oder Open Source Software gleichzusetzen. Die Hauptunterschiede beziehen sich

Elektronische Software Distribution

auf die Lizenzierungsform und werden im Kapitel 2.3 Software-Lizenzierung genauer betrachtet.

2.2.3 Freeware

Freeware ist wie Public Domain Software kostenlos erhältlich. Sie kann aber Nutzungseinschränkungen aufweisen, z.B. das nicht die gesamte Funktionalität vorhanden ist oder die Leistungsfähigkeit reduziert ist. Der Quellcode ist nicht verfügbar. Anwender erhalten ein *Black-Box*-Produkt, welches sie weder modifizieren, noch zu Lernzwecken nutzen können. Daher ist nur die unveränderte Weitergabe möglich und zulässig. Es gibt weder Support noch wird ein Besitzer eines Freewareprogramms gezielt mit Informationen versorgt. Der/die Entwickler verzichtet/n auf eine Vergütung. Allerdings gibt es die Möglichkeit, Anwendern die Bezahlung freizustellen. Diese Freeware wird dann als *Donationware* bezeichnet. Daneben existiert noch die so genannte *Cardware*, bei der der Autor um die Zusendung einer Postkarte bittet, als Anerkennung seiner Leistung.

2.2.4 Shareware und Varianten

Shareware bezeichnet allgemein eine nutzungsbeschränkte Softwareform, die erst nach einer Vergütung uneingeschränkt lauffähig ist. Die *Shareware Autoren Vereinigung* (SAVE) bezeichnet Shareware als „[...]Software, die - als Shareware-Version - nach den Vorgaben des Autors beliebig kopiert und an Dritte weitergegeben werden darf, ohne dass dadurch Urheberrechte verletzt werden.[...]“⁽²³⁾. Die Kopien dürfen nur kostenlos oder gegen eine Kopiegebühr in Höhe der anfallenden Kopierkosten weitergegeben werden. Dies soll auch der ursprüngliche Gedanke bei der Verteilung von Sharewareprogrammen gewesen sein. Als Begründer werden Jim „Button“ Knopf und Andrew Fluegelman benannt. Jim Knopf beschreibt, dass er ein Tool entwickelt hat, welches er an Freunde und Arbeitskollegen verteilte. Die Verbreitung des Programms war so enorm, dass er finanziell und zeitlich nicht mehr in der Lage war, alle Nutzer per Post mit Updates und Informationen zu versorgen. Daher beschloss er eine Meldung in das Programm einzubauen, die nach einer freiwilligen Bezahlung von 10 US Dollar fragte. All diejenigen, die der Bitte nachkämen, würden auf Jims Mailingliste gesetzt werden. Kurz darauf wurde er auf ein anderes Tool aufmerksam gemacht, dass ebenfalls eine solch ungewöhnliche Meldung ausgab. Die Autoren taten sich zusammen. Somit waren die beiden ersten Sharewareprogramme geboren⁷.

⁷ Aufgrund der Mailingliste besteht ein Support, der über Updates und Neuerungen informiert. Somit sind die Programme keine *Donationware*.

Elektronische Software Distribution

Im heutigen Kontext wird Shareware als *Prüf-vor-Kauf-Software* verstanden. Die Autoren beschränken die Nutzung auf einen bestimmten Zeitraum oder auf eine bestimmte Anzahl an Programmstarts. Evtl. Kunden sollen die Möglichkeit des Ausprobierens erhalten. Nach Ablauf der gesetzten Frist können die Funktionen oder das Programm selbst nicht mehr genutzt werden. Erst eine Registrierung und die damit verbundene Bezahlung ermöglicht eine weitere Nutzung des Programms. Zusätzlich können Funktionen eingeschränkt sein, die nur in einer Vollversion zur Verfügung gestellt werden sollen.

Es gibt verschiedene Möglichkeiten, Sharewareprogramme auszuliefern:

- als Vollversion, die mittels eines Freischaltmechanismus zur uneingeschränkten Nutzung verfügbar wird
- als Teilprodukt, welches nur ein separates Teilstück des ganzen Produktes ist
- nur bestimmte Basis-Features werden als Probe mitgegeben. Entscheidet sich ein Konsument für den Kauf, so werden ihm je nach Bezahlung weitere Features ausgehändigt. Ein Beispiel ist die *Game-Feature-Plattform* der 4FO AG (8)

Ist ein Programm nach Ablauf der Testperiode nicht mehr lauffähig, so spricht man von *Demoware*. Diese Variante, auch als *Try-and-Die* bezeichnet, soll nur eine Demonstration der Funktionalität des Programms liefern und potentielle Kunden zum Kauf bewegen. Der Begriff *Trialware* wird sowohl für *Demoware* und *Prüf-vor-Kauf*Software benutzt.

Obwohl es sich um Probeprodukte handelt, müssen auch sie ausreichend geschützt werden, um gegen Softwarepiraterie gesichert zu sein. Gerade Vollversionen, die nur durch einen Registrierungsschlüssel oder mit einem ähnlichen Mechanismus freigeschaltet werden müssen, sind ein beliebtes Angriffsziel von Hackern und Crackern (siehe Kapitel 3 Softwarepiraterie).

Es existieren darüber hinaus weitere Sharewarevarianten.

- Manche Programme enthalten einen so genannten *Nagscreen* (engl.: Nörgelbildschirm), der während der Benutzung des Programms immer wieder auf eine Registrierung drängt. Daher wird diese Variante als *Registerware* bezeichnet. Die Zeitabstände zwischen den Erscheinungen der Meldung werden dabei immer kürzer.
- Daneben steht die so genannte *Timeware*. Hier wird das laufende Programm nach einer bestimmten Zeitspanne angehalten und nur nach einem erneuten Start kann es wieder genutzt werden.
- Als *Aidware* bezeichnet man Sharewareprogramme, bei denen die Registrierungsgebühr, anstelle des Autors, an eine gemeinnützige oder karitative Organisation oder Einrichtung überwiesen wird.

Elektronische Software Distribution

- Ist die Funktionalität eines Programms dermaßen eingeschränkt oder der Programmablauf fehlerhaft, spricht man von *Crippleware*. Das Programm ist so stark „verkrüppelt“, dass ein Testen nicht mehr möglich und/oder sinnvoll erscheint.

2.2.5 Weitere Formen

Es existieren weitere Softwareformen. Eine der unangenehmsten ist die so genannte *Malware*. Darunter versteht man „böswillige“ Computerprogramme, die im Allgemeinen Schaden anrichten oder die Sicherheits- und Schutzfunktionen von Systemen untergraben bzw. umgehen sollen. Die Funktionalität dieser Programme ist in den meisten Fällen nicht fehlerhaft, sondern vom Autor gewollt. In diese Kategorie fallen Viren, Würmer, Trojanische Pferde, Backdoor-Programme und Spyware.

Dialer-Programme können nicht zu dieser Kategorie gezählt werden, da sie im ursprünglichen Sinne nicht als Schadprogramme fungieren. Ursprünglich waren sie als anonyme, elektronische Bezahlungsmöglichkeit für kostenpflichtige Inhalte gedacht. Erst wenn der Dialer eine Verbindung aufgebaut hat, erfolgt die Abrechnung über die Telefonrechnung. Die unseriöse Nutzung brachte Dialer-Programmen ihren schlechten Ruf ein. Sie wurden oftmals ohne Wissen des Nutzers installiert und versuchten bzw. bauten ohne Aufforderung Verbindungen zu teuren Mehrwertdiensten auf.

Adware bezeichnet eine Softwareform, die für Werbezwecke genutzt wird. Das beworbene Produkt wird dabei durch Werbebanner oder Werbe-Popups gepriesen.

Bookware bezeichnet die zu einem Buch oder Bedienungsanleitung auf einem physischen Träger beigefügte Software.

Der Begriff *Bananenware* bezeichnet in ironischer, negativer Weise mangelhafte Software, die an Konsumenten ausgeliefert wird, mit der Hoffnung, dass das Konsumentefeedback das Produkt „zur Reife“ bringen wird. Dies geschieht dann meist durch Updates.

2.3 Software-Lizenzierung

Software-Lizenzen können allgemein als Verträge über die Nutzungsrechte eines oder mehrerer Konsumenten bzgl. eines oder mehrerer Exemplare eines Softwareprodukts eines Herstellers bezeichnet werden. Lizenzen unterliegen dem Urheberrecht und beschreiben eine Überlassungsform. So sehen manche dieser Lizenzen vor, dass Konsumenten sich bereit erklären müssen, Bedingungen zu erfüllen und/oder auf Rechte zu verzichten, die ihnen vom Urheberrecht her zustehen. Andere beschränken weder die Nutzung, noch die Weitergabe.

Elektronische Software Distribution

Es existieren zahlreiche verschiedene so genannte Lizenzmodelle. In Unternehmen bspw. ist das Lizenzmodell eng an das Geschäftsmodell gekoppelt⁸. Sie lassen sich grob in zwei Kategorien einteilen, welche im Folgenden beschrieben werden. Allen gemein ist die Inanspruchnahme des Urheberrechts. Eine mit einem Produkt ausgelieferte Lizenz ist demzufolge ein konkretes Exemplar eines Lizenzmodells.

2.3.1 Free Software und Open Source Software License

Die Begriffe „Freie Software“ und „Open Source“ werden oftmals als Synonyme benutzt. Dennoch sind die zugrunde liegenden Ideologien von den dahinter stehenden Gemeinschaften über Jahre hinweg immer differenzierter betrachtet worden. Somit existieren zwei Lager, die trotz Unterschiede für ein gemeinsames Ziel eintreten: eine offene Softwareentwicklung.

Der Begriff „Freie Software“ steht für eine Philosophie, die den Freiheitsgedanken an oberste Stelle stellt. „Frei“ bedeutet hier nicht kostenlos⁹, sondern politisch, wirtschaftlich und gesellschaftlich ohne Hinderungen zugänglich. Diese Freiheit fußt auf vier Säulen:

1. Software darf ohne Einschränkungen genutzt werden
2. Quellcode freier Software ist verfügbar
3. Software darf ohne Einschränkungen und ohne Zahlungsverpflichtungen kopiert und weitergegeben werden
4. Software darf modifiziert und in veränderter Form weitergegeben werden

Die Idee ist eine offene und gemeinschaftsorientierte Softwareentwicklung. Die Mitarbeit ist durch das Interesse an einem Projekt motiviert und nicht oder selten gewinnorientiert. Jedes Mitglied einer Entwicklungsgemeinschaft ist dem anderen gleichgestellt. Damit Projekte nicht völlig undurchdacht weitergeführt werden, entsteht immer ein so genanntes „Core“-Team. Die Mitglieder beraten über zukünftige Entwicklungsschritte und gelten als erste Ansprechpartner bei Problemfragen. Sie entscheiden ebenfalls über Änderungen und Neu-Integrationen innerhalb eines Projekts. Große Projekte werden in einzelne Module geteilt, die von so genannten *Maintainern* in dem Sinne verwaltet werden, dass sie Fragen über die in einem Modul entwickelte Software beantworten und die Gemeinschaftsarbeit koordinieren.

„Freie Software“-Projekte werden im Allgemeinen niemals beendet, wie es bei proprietärer Software der Fall ist. Ständige Ideen und Entwicklungsdiskussionen durch Interessierte sorgen für immer neue Weiterentwicklung. Diese Innovationskraft ist wahrscheinlich die größte Stärke „Freier Software“. Zugleich fördert das Verlangen nach einem besseren Programm die Entwicklung verlässlicher Software. Das bedeutet, dass die Software von vielen Entwicklern

⁸ Geschäftsmodelle beschreiben die Verkaufsstrategie der Produkte an die Käufer

⁹ wie es Freeware ist

Elektronische Software Distribution

unterschiedlicher Kenntnisstufen auf Qualität getestet wird. Das Fehlen einer Organisationshierarchie und monetären Zwängen erleichtert oftmals Korrekturen und kann zu qualitativ höherwertiger Software, als es proprietäre sein kann, führen.

Die beiden ersten „Freie Software“-Lizenzen waren die MIT und BSD Lizenzen. Die MIT-Lizenz besagt lapidar, dass der Erhalt einer Kopie der Software die freie Verfügbarkeit und Weiterverbreitung garantiert¹⁰. Dies wird bei allen Kopien so verlangt. Die BSD-Lizenz ist ein wenig restriktiver. In ihrer Ursprungsform wird verlangt, dass ein von einer unter BSD-Lizenz entwickelten Software abgeleitetes Produkt darauf hinweist, dass Teile von der University of California, Lawrence Berkeley Laboratory entwickelt wurden. Darüber hinaus darf nur mit schriftlicher Erlaubnis der Name des Autors des Produkts angegeben werden. BSD-Lizenzen existieren heutzutage in vielen Variationen.

Die wohl bekannteste „Freie Software“-Lizenz ist die *GNU General Public License* (GPL). Unter der GPL entwickelte Software ist unveräußerliche Software, d.h. sie selbst als auch alle von ihr abgeleiteten Werke müssen „frei“ sein und dürfen keinerlei weiteren Restriktionen unterliegen oder diese hinzugefügt werden („[...]einmal frei, immer frei. [...]“ (32)). Das angewendete Prinzip wird als *Copyleft* bezeichnet. Dadurch werden die im Urheberrecht zur Privatisierung verwendeten Mittel zum Erhalt der Freiheit einer Software genutzt.

Eine weichere Form ist die *GNU Lesser General Public License* (LGPL). Sie bietet die Möglichkeit, unter GPL-Lizenz entwickelte Software mit proprietärer Software gemeinsam ausgeliefert zu werden, ohne gegen die GPL zu verstoßen.

Neben den genannten Lizenzen existieren noch eine Vielzahl weiterer, die hier aber nicht weiter beschrieben werden sollen, da dies nicht wesentlicher Bestandteil der Arbeit ist.

Der Begriff „Open Source“ wurde eingeführt, um das Problem des Wortes „Frei“ zu umgehen¹¹. „Quelloffenheit“ besagt, dass der Quellcode einsehbar ist. Das ist jedoch kein hinreichendes Kriterium, um die Software als frei zu bezeichnen. So können die Quellen offen gelegt, aber Modifikation und/oder Weitergabe untersagt sein. Der Fokus liegt somit mehr auf den technischen Möglichkeiten der Quelloffenheit als auf den eigentlichen ideologischen Zielen der „Freien Software“.

¹⁰ Der Verzicht auf alle Rechte bedeutet noch nicht, dass ein Werk der Allgemeinheit (*Public Domain*) zugeprochen wird

¹¹ Das englische Wort „free“ sorgte für einige Verständnisschwierigkeiten. Daher sollte durch das Wort „Quelloffenheit“ diese linguistische Ungenauigkeit entfernt werden

2.3.2 Non-Free / Non-Open Source License

Das Gegenstück zu „Freie Software“ bzw. „Open Source“ ist proprietäre Software. Dieser Begriff bezeichnet jedwede Art von Software, die mit irgendeiner Art von Restriktionen erworben werden. Hier finden die Mechanismen des Urheberrechts Anwendung. Das geistige Eigentum wird gesetzlich geschützt, indem Modifikationen und das Kopieren untersagt oder nur mit Erlaubnis der Hersteller legal sind. Selbst die Benutzung ist restriktiv. In der Wirtschaft wird nicht zwischen dem Quellcode und dem ausführbaren Programm unterschieden. Der Quellcode wird einfach nicht offen gelegt (*closed source*). So ist auch Freeware proprietäre Software.

Der Vertrieb proprietärer Softwareprogramme hat für Unternehmen das oberste Ziel der Rückführung der Investitionen und darüber hinaus Erwirtschaftung von Gewinnen. Die weitesten verbreitete Überlassungsform ist die fortwährende Lizenzierung. Der Erwerb eines Produkts gewährt Käufern dessen uneingeschränkte, immerdauernde Nutzung, einschließlich Support. Im Allgemeinen nicht enthalten sind Upgrades, welche größtenteils kostenpflichtig zur Verfügung gestellt werden. Für diese Art lizenzierter Software besteht das höchste Sicherheitsrisiko bzgl. Softwarepiraterie. Daher werden Vorkehrungen getroffen, die Programmnutzung exklusiv für eine bestimmte Anzahl legaler Nutzer zu erlauben. Gleichzeitig sollen die Anwender an illegalen Aktionen, z.B. unerlaubte Vervielfältigung, gehindert werden (siehe 3 Softwarepiraterie). Neue Lizenzmodelle für proprietäre Software sollen dieses Risiko ebenfalls minimieren.

Zwei dieser neuen Modelle basieren auf Verleihprinzipien. Das so genannten *Subscription License* Modell erlaubt es Konsumenten, sich für ein Produkt zu registrieren. Dadurch erhält dieser für eine festgelegte Zeitspanne den vollen Zugriff, Upgrades und Support. Die Registrierung kann immer wieder verlängert werden. Der Vorteil für den Konsumenten ist, dass er nur eine Leihgebühr bezahlt und nicht den vollen Preis für das Produkt und anfallende Upgrades. Der Anbieter ist natürlich darauf bedacht, durch ein gutes Produkt und kundenbefriedigende Dienstleistungen bestehende Registrierungen zu verlängern und eine breite Kundenbasis zu schaffen. Sollte ein Konsument nicht verlängern, so entfallen die Auslieferung neuer Upgrades und der Support. Das Produkt und alle erworbenen Updates dürfen weiterhin genutzt werden. Dies ist der Unterschied zum so genannten *Rental License* Modell. Hier kann ein Konsument das Produkt nur dann weiter nutzen, wenn er es erneut ausleiht.

Ein weiteres Modell ist die so genannten *Trialware License*. Wie schon weiter oben im Kapitel 2.2.4 Shareware und Varianten beschrieben, handelt es sich bei *Trialware* um eine zeit- oder nutzungsbeschränkte Produktversion.

Elektronische Software Distribution

Durch dieses *Prüf-vor-Kauf*Prinzip sollen potentielle Nutzer die Möglichkeit des Ausprobierens erhalten. Die Sicherung solcher Prüfprodukte ist für Unternehmen wichtig, um finanzielle Einbußen zu vermeiden.

Floating License bezeichnet ein Lizenzierungsmodell für eine begrenzte Anzahl von Nutzerlizenzen eines Produkts innerhalb eines Netzwerks. Die Anzahl an Lizenzen ist geringer als die Zahl aller potentiell möglichen Nutzer. Für jede gestartete Kopie des Produkts wird eine Lizenz in Anspruch genommen, bis das Maximum erreicht ist.

Alle Versuche, eine weitere Kopie zu starten ist nicht möglich. Die Lizenzen liegen auf einem separaten Lizenz-Server. Der Vorteil ist die Flexibilität innerhalb des Netzwerks. Das Programm kann auf jedem im Netzwerk teilnehmenden Computer gestartet werden, auf dem es installiert wurde. Somit ist die Lizenz nicht an einen lokalen Rechner gebunden, sondern kann von verschiedenen Nutzern auf verschiedenen Computern genutzt werden. Das Maximum sollte dabei mit Bedacht gewählt werden. Zu viele Lizenzen sind Geldverschwendung, zu wenige können die Entwicklungs- oder allgemeinen Geschäftsprozesse behindern.

Es existieren Lizenzmodelle, die eine Vermischung von proprietärer und „Open Source“ bzw. „Freie Software“-Lizenzen sind. Die *Sun Community Source License* (SCSL) ist solch eine Lizenz. Sun hat die Vorteile der „Open Source“-Entwicklung als Chance verstanden, um die eigenen Produkte zu verbessern. Die SCSL ist aufgrund einer von Sun verlangten Kompatibilitätsprüfung und -Anerkennung abgeleiteter Werke restriktiver, als es das „Open Source“-Modell zulässt. Damit fällt sie noch immer unter die Kategorie „Non-Open Source Software License“.

3 Softwarepiraterie

Der Begriff Softwarepiraterie bezeichnet die illegale Verbreitung und Nutzung copyright-geschützter Softwareprodukte. Für Unternehmen wie auch für Privatentwickler handelt es sich um großes, vornehmlich finanzielles Problem. Die Entwicklung von Softwareprodukten ist mit Kosten verbunden, die mit deren Größe und Komplexität steigen. Die Rückführung der Investitionen ist somit eine Notwendigkeit, um das Überleben eines Unternehmens oder einer Person sicher zu stellen.

Dieses Kapitel beschreibt das Problem der Softwarepiraterie. Anfangs wird auf Schäden und Konsequenzen für Hersteller und Konsumenten eingegangen, um darauf aufbauend eine Klassifizierung durchzuführen. Anschließend werden Möglichkeiten zur Vermeidung bzw. Behinderung von Softwarepiraterie beschrieben. Dabei werden sowohl technische als auch gesellschaftliche Ansätze betrachtet.

3.1 Schaden durch Piraterie

Der Austausch und die Modifikation von Programmen und deren Quellcodes war bis Anfang der 1980er eine gängige Praxis unter Entwicklern. Der Begriff „Freie Software“ existierte als solcher noch nicht, da Software an sich ein freies Gut war. Erst mit der Einführung von Geheimhaltungsverträgen und Lizenzen durch die Computerindustrie wandelte sich das Weltbild. Unternehmen hatten erkannt, dass sich mit Software Geld verdienen ließ und kontrollierten von nun an die Entwicklung, sowie die Regeln der Verteilung und Nutzung.

Der Begriff Softwarepiraterie bezeichnet die unautorisierte Vervielfältigung, Modifizierung, Benutzung und Verteilung proprietärer Software, sowohl in binärer Form als auch des Quellcodes. Es handelt sich dabei um eine Verletzung des Urheberrechts und um Diebstahl geistigen Eigentums. Zusätzlich gibt es einen beträchtlichen finanziellen Schaden. Dazu werden jährlich Piraterie-Studien durch die *Business Software Alliance* (BSA¹²) in Kooperation mit der *International Data Corporation* (IDC) durchgeführt. Die BSA beschreibt sich selbst als Non-Profit Gesellschaft, die sich dem Kampf gegen die Softwarepiraterie verschrieben hat. Ihre Piraterie-Studien sollen den weltweiten Piraterie-Verbreitungsgrad und die jährliche Piraterierate empirisch mittels Interviews und Recherchen von Marktanalysten in den jeweiligen Ländern und Landstrichen ermitteln. Die BSA ist von großen Softwarefirmen wie Adobe, Apple, IBM, Microsoft und vielen anderen gegründet worden. Diese Unternehmen mussten alle schon große finanzielle Einbußen durch Softwarepiraterie hinnehmen.

¹² <http://www.bsa.org>

Softwairpiraterie

Daher verwundert es nicht, dass die BSA ihre Prioritäten und Ziele entsprechend definiert. Ob sie daher als wirkliche Non-Profit-Organisation verstanden werden darf, bleibt hinter diesem Hintergrund fraglich.

In der im Mai 2006 vorgelegten Studie wurde eine weltweite Piraterierate von 35% ermittelt. Diese Zahl bezieht sich nur auf finanzielle Aspekte. Konkret bedeutet dies, dass im Jahr 2005 weltweit über 60 Milliarden U.S. Dollar für Software ausgegeben wurden. Allerdings wurde laut IDC Software in Höhe von 94 Milliarden U.S. Dollar installiert. Sollte diese Berechnung in etwa stimmen, haben Firmen weltweit einen Gesamtverlust von 34 Milliarden U.S. Dollar hinnehmen müssen. Dass dies nicht ganz der Wahrheit entspricht ergibt sich durch eine genauere Betrachtung der verwendeten Methode zur Berechnung der Piraterierate. Die folgende Abbildung 1 zeigt verwendete Bestimmungsmethode, entnommen aus der Pirateriestudie für 2005 (siehe (33)).

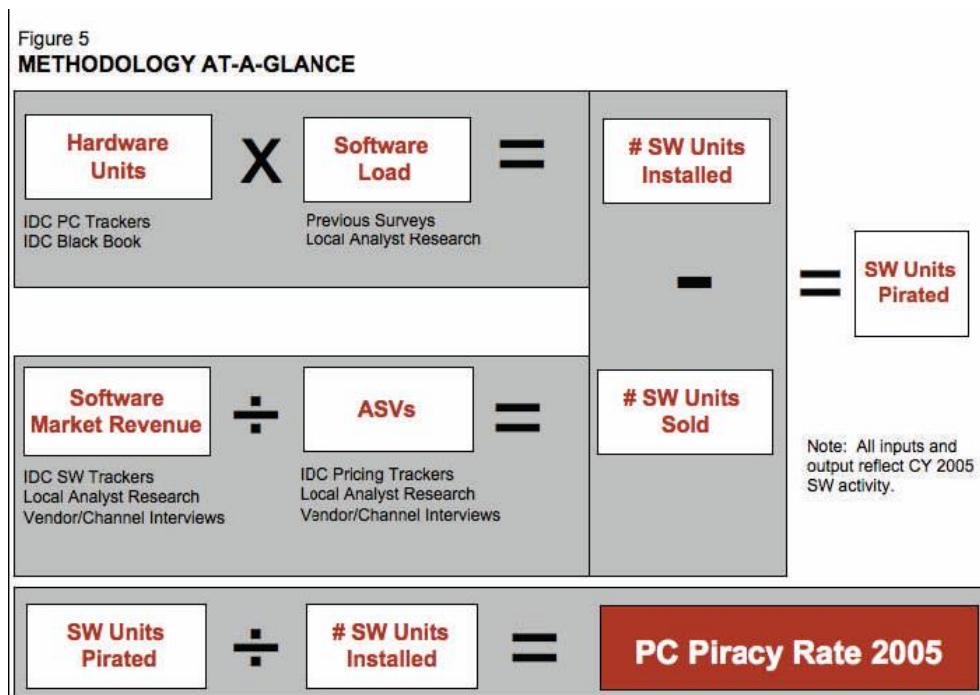


Abbildung 1 Methode zur Berechnung der Piraterierate nach IDC

Es wird die Gesamtmenge der installierten Software von der Gesamtmenge verkaufter Software subtrahiert. Das Ergebnis ist dann die Menge an Software, die im Jahr 2005 der Piraterie zum Opfer gefallen ist. Ein Element zur Berechnung der Gesamtmenge installierter Software ist der so genannten *Software Load*. Das ist die Menge an installierter und/oder vorinstallierter Software auf PCs im Jahr. Darunter fallen ebenfalls Programme, die als Freeware, Shareware oder Open Source lizenziert sind. Solche Software als illegale Piraterieprodukte zu deklarieren ist eine Verzerrung der Wirklichkeit.

Softwarepiraterie

Dies hat zur Folge, dass die zur Berechnung herangezogenen Werte schlichtweg falsch sind und die Menge illegaler Software nicht so hoch ist, wie die Studie es weiß machen will.

Das die Studie jedoch aussagekräftige Ergebnisse liefert, wird bei detaillierter Betrachtung sichtbar. Die Piraterierate ist in den entwickelten Ländern zum Großteil relativ niedrig, im Schnitt um die 30 %, während in Entwicklungs- und Schwellenländern der Anteil an illegaler Software bis zu 90 % ausmachen kann. Aufgrund des Verhältnisses zwischen der Anzahl an Industrie- und Schwellenländern beträgt die weltweite durchschnittliche Piraterierate über 60%. Aufgrund der riesigen IT-Märkte in den Erste Welt Ländern wie USA, Kanada, Japan und der Europäischen Union beläuft sich der finanzielle Verlust auf ungefähr 22 Milliarden U.S. Dollar. Das entspricht in etwa Zwei-Drittel des weltweiten Gesamtverlusts. Absolut führend sind dabei die USA mit einem Verlust von knapp sieben Milliarden U.S. Dollar. Erst mit großem Abstand folgt China mit knapp vier Milliarden U.S. Dollar.

Der finanzielle Verlust beträgt in den Entwicklungs- und Schwellenländern zwar „nur“ um die 12 Milliarden U.S. Dollar, allerdings ist legale Software nur für sieben Milliarden U.S. Dollar verkauft worden. Hier zeigen sich ein enormer Missstand, sowie das Verhältnis und Verständnis zum Begriff des geistigen Eigentums.

Die IDC hat in einer weiteren Studie vorgeschlagen, die Piraterierate innerhalb der nächsten vier Jahre um 10 % zu senken. Dies würde in etwa 2,4 Million neuer Arbeitsplätze und über 400 Milliarden U.S. Dollar an Wirtschaftszuwachs weltweit bedeuten (34).

3.2 Kategorisierung von Piraterie

Softwarepiraterie bedeutet immer, dass auf irgendeine Art und Weise die Lizenz missachtet wird, als auch ein evtl. vorhandener Kopierschutz ausgehebelt wurde. Man kann demnach Piraterie in verschiedene Kategorien einteilen.

Eine weit verbreitete Methode der Softwarepiraterie wird als *Softlifting* bezeichnet. Ein Konsument erwirbt eine legale Kopie eines Programms. Entgegen der Lizenzbestimmungen wird diese Kopie auf mehreren Computern installiert. Dies betrifft nicht nur Privatanwender, die an Freunde und Bekannte Kopien verteilen. In Unternehmen wird diese Art ebenfalls häufig genutzt, um Kosten zu sparen.

Ebenso weit verbreitet ist die unautorisierte Verteilung urheberrechtlich geschützter Software über technische Kommunikationskanäle wie dem Internet oder Tauschbörsen. Diese simple, als *Filesharing* benannte Methode kann erweitert werden. Dazu werden Kopien derart designed und hergerichtet, als würde es sich um eine legitime Kopie handeln. Dies wird als

Softwairpiraterie

Software Counterfeiting bezeichnet. Solche Software wird entweder verkauft oder nach dem *Rental License* Modell vermietet (siehe 2.3.2 Non-Free / Non-Open Source License).

OEM unbundling beschreibt eine Methode, mit der Software verteilt wird, die eigentlich zusammen mit einer bestimmten Hardware ausgeliefert werden sollte.

3.3 Schutz vor Piraterie

Das Ziel eines jeden softwareherstellenden Unternehmens ist der Verkauf lizenzierter Kopien ihrer Produkte. Dabei handelt es sich im Allgemeinen um proprietäre Software, die heutzutage oft zusätzlich als Trialversion vorliegt. Die Rückführung der Entwicklungskosten und darüber hinaus Gewinne zu erzielen sind die obersten Ziele. Damit diese finanziellen Notwendigkeiten innerhalb eines marktwirtschaftlichen Systems durchgesetzt werden können, müssen Unternehmen sich und ihre Produkte vor der Gefahr der Piraterie schützen. Das folgende Kapitel beschäftigt sich mit den technischen Realisierungsmöglichkeiten, während das darauf folgende sich mit einem sozial-gesellschaftlichem Strategiekonzept auseinandersetzt.

3.3.1 Technische Möglichkeiten

Die Durchsetzung zur Einhaltung von Lizenzbestimmungen und Schutz vor Missbrauch mittels technischer Methoden wird als Kopierschutzverfahren bezeichnet. Diese müssen verschiedenste Anforderungen erfüllen.

Für Konsumenten ist die Transparenz das wahrscheinlich wichtigste Kriterium. Kein Nutzer würde es verstehen, wenn aufgrund eines Kopierschutzmechanismus eine legal erworbene Kopie nicht ablauffähig wäre. Demzufolge muss ein qualitativ hochwertiger Kopierschutz mit vielen verschiedenen Hard- und Softwarekonfigurationen operieren können.

Für Softwarehersteller ist die Robustheit gegen Attacken ein sehr wichtiges Kriterium. Es ist praktisch nicht möglich, bei den heutigen offenen Computer-Architekturen einen unüberwindbaren Kopierschutz zu entwickeln. Es stellt sich daher die Frage, wie robust der Mechanismus sein soll, d.h. wie lange die Schutzfunktionen Angriffen standhalten können. Die Spieleindustrie benötigt extrem robuste Schutzverfahren, während es für andere Anwendungen sinnvoller sein kann, einen simpleren Mechanismus zu integrieren, um das Produkt gegen Angriffe durchschnittlicher Cracker zu schützen.

Neben der Robustheit ist die Flexibilität eines Kopierschutzes wichtig. Flexibel bedeutet, dass ein Hersteller aus verschiedenen Schutzschemata wählen kann. So kann ein Produkt als Vollversion mit Aktivierungsfunktion ausgeliefert werden, ein anderes als Shareware mit eingeschränkter Funktionalität und limitierten Programmstarts. Das bedeutet allerdings auch, dass Trialversionen genauso gut geschützt werden müssen, wie das eigentliche Produkt selbst.

Softwairpiraterie

Die Auslieferung als Shareware, die nur durch einen Registrierungsschlüssel zur Vollversion wird, ist dementsprechend genauso stark gefährdet, wie das eigentliche Produkt. Für einen Hersteller ist zudem der finanzielle Verlust zu beklagen.

Kopierschutzverfahren können auf Hardware- als auch auf Softwareebene existieren. Der größte Vorteil hardware-basierter Verfahren ist der hohe Grad an Schutz, den sie gegenüber software-basierten Mechanismen erreichen können. Allerdings müssen die entsprechenden Komponenten mit dem Produkt ausgeliefert werden, was zusätzliche finanzielle Kosten verursacht. Genau diese Nachteile sind die große Stärke software-basierter Lösungen. Sie sind sehr viel kostengünstiger zu entwickeln und werden in den Programmen integriert. Daher sind sie für den Einsatz in ESD-Systemen hervorragend geeignet. Die große Schwäche der meisten software-basierten Kopierschutzverfahren ist das Problem der so genannten *class breaks* (siehe (19)). Dieser Begriff tituliert ein generisches Fehlverhalten eines Schutzmechanismus, der auf irgendeine Art überwunden wurde. Generisch, da es sich nicht nur auf ein spezielles System bezieht, sondern sämtliche Software, die mittels eines überwundenen Verfahrens vor Piraterie gesichert werden sollte, potentiell gefährdet ist und unerlaubt kopiert werden könnte.

3.3.1.1 Verschlüsselung –Herzstück bestehender Kopierschutzverfahren

Jedes vernünftige Kopierschutzverfahren basiert auf Kryptographie. Ein Produkt oder der verwendete Kopierschutz werden in verschlüsselter Form an die Konsumenten weitergegeben. Es existiert entsprechend ein Mechanismus zum Entschlüsseln, damit das Programm ausgeführt werden kann. Genau dies ist der Schwachpunkt aller bisherigen software-basierten Kopierschutzmechanismen. Heutige Hardware-Architekturen von Computern sind „offen“, d.h. sämtliche Software wird von der CPU ohne Restriktionen verarbeitet¹³. Dazu zählen alle Operationen und Daten zur Entschlüsselung. Dies betrifft sowohl den Programmcode und Algorithmen zur Entschlüsselung, als auch die benötigten Schlüssel. Selbst wenn der bzw. die Schlüssel weiter geheim gehalten werden kann bzw. können, liegen die Programmdateien in entschlüsselter Form vor. Ein Hacker kann jederzeit diese Daten aus dem Speicher kopieren. Das Problem ist, dass es zurzeit keine Möglichkeit gibt, die Daten, nachdem sie entschlüsselt wurden, geschützt zu halten. Dies ist nur durch Modifikationen der Hardware zu erreichen, was im Kapitel 3.3.1.6 Hardware-basierter Schutz näher erläutert wird.

Die Verschlüsselung eines Programms findet nach dessen Kompilierung statt. Der Schlüssel zum Entschlüsseln wird innerhalb des Programms versteckt bzw. der Algorithmus zu dessen Berechnung zur Laufzeit.

¹³ Die CPU kann nur zwischen privilegiertem und nicht-privilegiertem Ausführungsmodus unterscheiden

Softwarerpiraterie

Letztere Taktik wird verwendet, um automatische Angriffe durch so genannte *Unpacker*-Programme zu verhindern oder extrem zu erschweren. Solche Programme „kennen“ die meisten Verschlüsselungsalgorithmen und versuchen, den verwendeten Schlüssel zu finden. Das Ergebnis ist die originale ausführbare Programmdatei ohne die Verschlüsselung.

3.3.1.2 Erweiterte Schutzkonzepte

Kryptologische Verfahren bilden die Basis für alle guten Kopierschutzverfahren. Es existieren darüber hinaus weitere Konzepte, um Programme gegen Piraterieangriffe zu schützen.

Mehrerer dieser Mechanismen werden unter dem Begriff *Anti-Reversing*-Techniken¹⁴ zusammengefasst. Diese Techniken sollen *Reverse Engineering* Attacken erschweren oder sogar verhindern. Unter *Reverse Engineering* versteht man Methoden zur Untersuchung und zum Herausfinden von Konzeptionen und Realisierungen von Programmen. Diese Programme liegen als *Black-Box* vor und geben keinerlei Hinweise auf ihren Aufbau und Abläufe. Der Einsatz von *Reverse Engineering* Techniken ist einerseits notwendig, um bspw. Kompatibilitätsprobleme zwischen Softwareprodukten zu lösen oder Sicherheitsüberprüfungen durchzuführen. Andererseits werden diese Techniken ebenfalls zum *Cracken* verwendet, um gegen Kopierschutzverfahren anzugehen.

Anti-Reversing-Techniken sind aktive Schutzmechanismen und sollen *Reversern* dermaßen viel Zeit und Aufwand abverlangen, dass diese ihre Bemühungen einstellen und somit zur Aufgabe gezwungen werden. Allerdings können die eingesetzten Mechanismen unterschiedliche Auswirkungen auf die zu schützenden Programme haben, z.B. wird die Programmgröße erhöht, die Zuverlässigkeit der Ausführung wird herabgesetzt, die Ausführung benötigt mehr Speicherplatz, etc. Trotz alledem ist der Einsatz bei vielen proprietären Produkten sinnvoll, um die Belange der Hersteller technisch durchzusetzen.

Eine der wirkungsvollsten Mechanismen stellt die so genannte *Code Obfuscation* (*engl.*: Code-Vernebelung) dar. Es handelt sich um eine Transformation einer ausführbaren Programmdatei in ein Äquivalent. Das Code-Layout und die Programmorganisation werden umgestaltet und gewinnen an Komplexität. Die Funktionalität des transformierten Programms ist (nahezu) identisch mit der des Originalprogramms. Idealerweise wird die Transformation mittels eines *Obfuscator*-Programms automatisch durchgeführt, das in der Lage ist, den gesamten Aufbau einer ausführbaren Datei umzugestalten. Der Quellcode bleibt davon unberührt.

¹⁴ *Reversing* ist ein künstliches Wort, das als Kurzform für *Software Reverse Engineering* eingesetzt wird. Es wird im Buch von Eldad Eilam eingeführt (19)

Softwarerpiraterie

Es ist natürlich möglich, *Obfuscation* auf Quellcodeebene einzusetzen, um die Leserlichkeit für Menschen herabzusetzen. Nach (21) sind vier Kriterien für die Qualität einer Transformation maßgeblich:

- die Wirkungsstärke, die den Grad an hinzugefügter Komplexität ausdrückt
- der Grad an Unverwüstlichkeit, d.h. die Schwierigkeit, eine Transformation zu entkräften, bspw. durch so genannte *Deobfuscator*-Programme
- der Verborgenhetsgrad, also in wie weit die Transformationen mit dem eigentlichen Programmcode harmonisch verschmelzen
- die Kosten, d.h. der hinzugekommene Overhead und die sich ergebenden Seiteneffekte

Obfuscation ist auf verschiedene Formate anwendbar und sinnvoll. Die Transformation bietet sich bei Software an, die in binärer Form für eine spezifische Prozessorarchitektur vorliegt, wie Intels *IA-32 Maschinen Code*. Die Einführung von plattform-unabhängigen Formaten wie Java-Byte-Code oder für Microsofts .Net-Plattform macht *Obfuscation* zu einer Notwendigkeit. Diese Formate sind leicht in Quellcode zurück zu transformieren, da sie viele Informationen über die im Quellcode erdachten Strukturen und Abläufe enthalten. Dies kann mittels eines *Decompilers* durchgeführt werden. Die Dekompilation von byte-code-basierten Softwareartefakten ist ein relativ einfacher Prozess, im Gegensatz zur Rückführung nativen Maschinencodes. Bei letzterem handelt es sich um einen sehr schwierigen und anfälligen Prozess, der durch einfache *Obfuscation*-Transformationen zu Fehlverhalten gezwungen werden kann (19).

Eine ebenfalls sehr effektive *Anti-Reversing* Maßnahme sind so genannte *Anti-Debugger*-Techniken. Zusammen mit Kryptographie handelt es sich um einen sehr starken Schutzmechanismus, der trotzdem vorsichtig genutzt werden sollte. Verschlüsselte Programme können nur ausgeführt werden, indem sie zur Laufzeit entschlüsselt werden. *Reverser* werden daher gezwungen, einen Debugger zu verwenden, um den Code analysieren zu können. Der Einsatz von Techniken zur Erschwerung dieses Prozess, so simpel sie auch sein mögen, bringt viele *Reverser* zur Aufgabe ihrer Versuche, da sie keine Tricks kennen, um diese Mechanismen zu umgehen. Debugger zielen auf Programmier-Hochsprachen ab der dritten Generation, wie C/C++, und sind an eine Programmiersprache gebunden.

Daneben existieren so genannten *Disassembler*. Ihre Nutzung zielt darauf ab, aus vorhandenem Maschinencode Assembler-Code zu generieren. Die Aufgabe besteht darin, den Disassembler mit Fehlinformationen zu versorgen und somit inkorrekten Code generieren zu lassen.

Softwairpiraterie

Ein weiteres Schutzkonzept ist das so genannte *Software-Watermarking* (22). Es handelt sich hierbei weniger um einen Schutz, sondern vielmehr um ein Verfahren zur Legalitätsbestimmung. Das Prinzip ist dem in der Musik- und Filmindustrie eingesetzte Verfahren ähnlich. Dabei werden dem eigentlichen Produkt verschlüsselte Informationen hinzugefügt, durch die ein Autor sein Werk und legal erworbene Kopien identifizieren kann. Wird für jede Kopie ein neues Wasserzeichen erzeugt, wird der Begriff *fingerprint* verwendet. Die Qualität eines Wasserzeichens bzw. Fingerabdrucks kann an zwei Eigenschaften gemessen werden. Zum einen kann die Stärke herangezogen werden. Die Berechnung wird durch drei weitere Faktoren bestimmt. Das sind die Quantität an hinzugefügten Daten (Datenrate), die Unsichtbarkeit und die Abwehr möglicher Angriffe. Zum anderen ist der Schutz gegen verschiedene Angriffstypen ein Maß zur Bestimmung der Qualität. Dieses Thema soll hier nicht weiter eingegangen werden. Für detaillierte Informationen siehe (22).

3.3.1.3 Serial-Number

Bei der *Serial-Number* handelt es sich wahrscheinlich um den am weitesten verbreiteten Kopierschutz. Das Prinzip ist relativ simple. Dem Produkt wird bei seiner Auslieferung auf irgendeine Art und Weise eine alphanumerische Zeichenkette mitgegeben, z.B. auf der Verpackung oder dem Trägermedium. Der Nutzer wird während der Installation dazu aufgefordert, diese Zeichenkette einzugeben. Mittels eines geheimen Algorithmus wird die Eingabe validiert. Damit soll die Legalität dieser Kopie sichergestellt werden. Bei Bestätigung der Eingabe als korrekt, wird das Programm weiter installiert und im System registriert.

Der Grad an Schutz vor Piraterie ist durch den alleinigen Einsatz einer *Serial-Number* gering. Die Software muss nicht einmal in verschlüsselter Form vorliegen, was den Schutzgrad weiter senkt. Das Aufspüren des Einstiegspunkts der Abfrage ist dann umso einfacher und hätte zur Folge, dass der Mechanismus einfacher überwunden würde. Zudem können sich Nutzer gegenseitig ihre Zeichenketten verraten oder, für einen Hersteller noch schrecklicheres Szenario, ein einzelner Nutzer veröffentlicht im Internet oder in Tauschbörsen ganze Listen mit korrekten *Serial-Numbers*. Die Ausbreitung an illegal installierten und lauffähigen Kopien wäre immens. Solange Konsumenten nicht auf den Support angewiesen sind und die Homepage des Herstellers kontaktieren, wird dieser keine Kenntnisse über den Verbreitungsgrad seiner Software haben.

Softwairpiraterie

3.3.1.4 Produktaktivierung

Die Produktaktivierung stellt eine Verbesserung des Schutzes gegenüber der *Serial-Number* dar. Die Installation erfolgt wie zuvor beschrieben. Der Unterschied ist, dass nach der einfachen Validierung durch das Programm zusätzlich eine weitere Identifikationsnummer generiert wird. Diese identifiziert den Computer auf dem die Software installiert wird bzw. wurde eindeutig. Der Nutzer muss auf irgendeine Art den Hersteller kontaktieren, um die generierte Identifikationsnummer validieren zu lassen. Eine Antwort wird zurückgeliefert, die in den Installationsprozess integriert bzw. zu einem anderen Zeitpunkt validiert wird. In (8) wird als Beispiel das Aktivierungsszenario von Microsoft beschrieben.

Technische Grundlage ist das so genannte *challenge response* Prinzip, auch als *Handshake* bezeichnet. Es wird innerhalb von Netzwerken zur Authentifizierung teilnehmender Nutzer und/oder Computer genutzt (mehr dazu, siehe (35)).

Für Software-Hersteller hat ein solcher Schutzmechanismus den Vorteil, dass durch eine erzwungene Kontaktierung ein einfacher *Serial-Number*-Austausch nicht mehr genügt, um eine ablauffähiges Programm zu installieren und zu nutzen. Die unüberprüfte Verbreitung der Zeichenketten kann reduziert werden. Dazu kommt noch die Übermittlung Nutzer- bzw. Computerspezifischer Daten, zur genauen Identifizierung eines Rechners. Eben diese Daten sind es, die von Datenschützern als kritisch angesehen werden. Da eine Produktaktivierung aber keine personenbezogenen Daten heranzieht, ist sie ein zulässiges Instrument zur Vermeidung von Softwairpiraterie. In (8) werden noch weiterführende Informationen zu diesem Thema beschrieben.

Obwohl es sich hierbei um einen verbesserten Schutzmechanismus handelt, kann auch er umgangen werden. Beispielsweise reichte bei der Produktaktivierung von Microsofts Betriebssystem Windows XP die Trennung des Netzkabels des Installationsrechners. Beim *Release Candidate 1* konnten Einträge in Systemdateien die Aktivierung aushebeln (45).

3.3.1.5 Software als Service

Die heutigen System-Architekturen und Infrastrukturen, sowie moderne Softwarepakete ermöglichen eine spezielle Form des Softwareschutzes. Ein Anbieter stellt nur noch die Funktionalität eines Programms über Schnittstellen als Dienst zur Verfügung. Konsumenten melden sich beim Dienst an und können ihn für die spezifizierten Aufgaben nutzen. Dies geschieht meist transaktional. Nach Vollendung der Aufgabe erhält der Konsument ein Ergebnis in irgendeiner Form zurück, beispielsweise eine Meldung über die erfolgreiche Durchführung der Anfrage, eine erwartete Zeichenkette, etc. Da niemand außer dem Anbieter des Dienstes

Softwairpiraterie

Zugriff auf das Programm hat, kann es von niemand anderem kopiert oder verbreitet werden. Dies ist der wahrscheinlich effektivste technische Schutzmechanismus. Dieses Modell ist allerdings auf server-basierte Software beschränkt und für viele Anwendungen nicht zweckmäßig. Daher wird diese Art nicht als wirklicher Kopierschutz angesehen. Das Konzept ist schon von verschiedenen Unternehmen als so genannte Web Services (WS) realisiert worden. Das in Kapitel 4 vorgestellte PotatoSystem (PS) benutzt Web Services, um für ausgewählte Nutzer einige Dienste anzubieten.

3.3.1.6 Hardware-basierter Schutz

Hardware-basierte Kopierschutzverfahren sind im Gegensatz zu Softwarelösungen schwerer zu überwinden. Es existiert eine zusätzliche Hardware-Komponente, die zu einem Programm mitgeliefert wird. Diese als *Dongle* bezeichnete Komponente ist ein zusätzlicher Authentifizierungsmechanismus. Ein *Dongle* ist ein Chip, der an eine der externen Schnittstellen eines Computers gesteckt wird, z.B. USB-Port.

Der Schutzgrad ist auch hier abhängig von der Implementierung des Mechanismus. Da es sich um eine Hardwarekomponente handelt, wird die Existenz eines Dongles nur über Treiberanfragen des Programms ermöglicht. Die trivialste Lösung ist, dass der Dongle zu Programmbeginn nur gefunden werden muss. Dieser Fall ist einfach zu attackieren, da nur diese Prüfung zu unterbinden ist. Eine verbesserte Variante basiert wiederum auf Kryptographie. Das Programm wird verschlüsselt ausgeliefert. Der/Die Schlüssel zur Entschlüsselung sind innerhalb des Dongles gespeichert. Das Programm muss nicht nur auf Existenz des Dongles prüfen, sondern benötigt darüber hinaus die Informationen im Innern. Obwohl es sich hier um eine verbesserte Schutzvariante handelt, ist sie ebenfalls stark anfällig. Das Problem ist wiederum, dass das Programm entschlüsselt im Arbeitsspeicher vorliegt und nach Start von dort entnommen werden kann, um eine neue, unverschlüsselte ausführbare Programmdatei zu erzeugen. Selbst der Einsatz mehrerer Schlüssel und die Aufspaltung des Programms in mehrere Programmeinheiten, die alle einen unterschiedlichen Schlüssel benötigen, ist keine Lösung. Es muss nur ein Dongle-Emulator erschaffen werden, um das anfragende Programm mit den gewünschten Informationen zu versorgen. Das beansprucht zwar mehr Zeit und Aufwand, ist aber durchaus realisierbar.

Es gilt dabei zwischen Einzelplatz- und Netzwerk-Dongles zu unterscheiden. Letzterer wird an den Port eines als Dongle-Server fungierenden Computers angebracht. Er enthält die Zugriffs-Informationen des vereinbarten Lizenzvertrags, z.B. Anzahl gestarteter Kopien innerhalb des Netzwerks.

Softwairpiraterie

Eine weitere Möglichkeit ist das Binden einer Software an ein spezifisches Gerät, bspw. wird die zu einem DVD-Brenner mitgelieferte Brennsoftware an dieses mittels der Seriennummer gebunden. Der Start des Programms erfolgt nur, wenn diese Komponente vorhanden ist und eine Validierungsüberprüfung stattgefunden hat. Dies ist zum einen kein zuverlässiger Mechanismus, da teilweise legitime Systeme als illegal ausgewertet werden, zum anderen ist bei einem Hardware-Update die Programmnutzung ausgeschlossen. Dies ist natürlich ein gutes Mittel, um Kundschaft zu vertreiben oder gar nicht erst zu bekommen. Ein findiger *Reverser* kann diesen Mechanismus umgehen.

3.3.1.7 Trusted Computing

Einen ganzheitlichen Ansatz verfolgt die *Trusted Computing Group* (TCG)¹⁵ mit dem Konzept des *Trusted Computing* (TC). Das Ziel ist die Definition und Entwicklung von Spezifikationen für hersteller-neutrale, sichere Plattformen. Die Motivation basiert auf den erhöhten Schutzbedürfnissen der Industrie, ihre Geschäftsdaten gegenüber verschiedenen Angriffen und Spionage zu schützen. Die steigende Konnektivität und die damit einhergehende Vernetzung steigert ebenfalls die Möglichkeit eines Angriffs auf die Systeme eines Unternehmens, beispielsweise durch Viren oder Trojaner. Die Unzulänglichkeiten von reinen software-basierten Schutzmechanismen wie SSL, VPN oder IPsec sollen durch die Spezifikation einer so genannten *Trusted Platform* (TP) überwunden werden. Eine TP muss zum einen in der Lage sein, Speicherbereiche exklusiv an vertrauenswürdige Kommandos zu vergeben (*protected capabilities*). Des Weiteren müssen die im System vorhandenen Informationen als korrekt zu evaluieren sein (*attestation*). Schließlich muss die Integrität eines Systems durch spezielle Messverfahren und –Werte gesichert sein (*integrity measurement, storage and reporting*). Die Basis bildet das Konzept der *Roots-of-Trust*. Dabei handelt es sich Hardwarekomponenten, die als vertrauenswürdig eingestuft werden müssen, da sie den Gesamtsicherheitszustand eines Systems evaluieren. Standardmäßig existieren drei zu unterscheidende „Wurzeln“. Die *Root-of-Trust for measurement* (RTM) zum Messen und Aufzeichnen von sicherheitsrelevanten System- und Integritätszuständen, die *Root-of-Trust for storage* (RTS) zum Speichern der Zustände und die *Root-of-Trust for reporting* (RTR) zum Wiedergeben. Die RTM wird auf verschiedene Komponenten bestehender Hardware abgebildet, während die RTR und RTS auf einen neuartigen Prozessor namens *Trusted Computing Module* (TPM) abgebildet werden. Das TPM ist das Herzstück einer TP. Es handelt sich um einen 8 bit und 33MHz RISC-Prozessor.

¹⁵ TCG ist ein Zusammenschluss großer Industrieunternehmen zu einer Non-Profit-Organisation. Sie löste die *Trusted Computing Platform Alliance* (TCPA) ab, adaptierte ihre Werke, um sie weiter zu führen

Softwarerpiraterie

Dieser Prozessor hält in seinem Innern verschiedene kryptographische Schlüssel, Zertifikate und Passwörter. Die RTM-Werte werden hier abgelegt. Jeder TPM ist einzigartig und ist mit der Platine eines PCs verlötet. Seine Entfernung würde die Vernichtung sämtlicher Daten bedeuten, was nicht heißt, dass der Computer nicht mehr lauffähig wäre. Einzig die auf dem TPM basierenden Applikationen würden nicht mehr funktionieren. Die aktuelle Spezifikation ist Version 1.2 Revision 85.

Das TPM soll als Grundlage sicherer Plattformen dienen. Der Zugriff auf die Funktionalitäten des Prozessors wird über ein Software-Interface namens *TCG Software Stack* (TSS) geregelt. Einerseits sollen auf dem TPM-basierende Applikationsentwicklungen erleichtert, andererseits Plattform-Interoperabilität erreicht werden.

Es existieren noch weitere Entwicklungsgruppen, wie die *Infrastructure Work Group*, die sich mit der Integration von TCG-Plattformspezifischen Spezifikationen in Internet- und Unternehmensinfrastrukturen beschäftigen, oder die *Mobile Phone Work Group* zur Adaption der TCG-Konzepte auf mobile Endgeräte. Für tiefergehende Informationen über alle vorgestellten *Trusted Computing* Konzepte und Spezifikationen soll an dieser Stelle auf die Homepage der TCG (37) verwiesen werden.

Es existieren schon einige Implementationen von IBM, Intel und teilweise von Microsoft. IBM bietet einige Produktreihen optional mit TCPA- und TCG-konformer Hardware-Technologie und IBM-spezifischer Software an. Intel entwickelte die Technologie unter dem Namen *LaGrande Technology* (LT) und arbeitete dabei vermutlich mit Microsoft zusammen. Microsoft entwickelt dabei ein Konzept namens *Next-Generation Secure Computing Base* (NGSCB), früher unter dem Namen Palladium bekannt. Die spezifizierten Sicherheitskonzepte sollen dabei in ein neues Betriebssystem integriert werden. Hier soll nicht weiter auf die Konzepte eingegangen werden. Das „Bundesamt für Sicherheit in der Informationstechnik“ bietet einen guten Ausgangspunkt, um sich detaillierter über die vorgestellten Mechanismen zu informieren (42).

Die starke Ausrichtung auf die Schutz- und Sicherheitsmechanismen findet nicht nur Zuspruch. In (8) wird Prof. Bill Caelli zitiert, der das von Microsoft in wahrscheinlicher Zusammenarbeit mit Intel bemühte Konzept als einen Versuch zur Kontrollausübung und -übernahmen von Anwender-Computersystemen kritisiert. Die Freiheit eines Anwenders wird damit derartig reduziert, dass es wohl dazu kommen könnte, dass freie verfügbare Software nicht mehr installier- und ausführbar sein wird. Die Nutzung von TC durch ein Programm ist nur mittels eines Zertifikats der TCG möglich, welche sich die Open-Source-Gemeinde aus finanzieller Sicht nicht leisten können wird.

Softwarepiraterie

Daneben existieren ernste Sicherheitsbedenken. Beispielsweise wird in (41) berichtet, dass der verwendete Hash-Algorithmus SHA-1 zur Identitätsprüfung, digitales Signieren und berechnen sicherer Bootsequenzen angeblich zu Beginn des Jahres 2005 gebrochenen wurde und sich die Angriffsmöglichkeiten seitdem erleichtert haben. Die in den Spezifikationen der TCG vorgeschlagenen Schutzkonzepte seien somit nicht mehr sicher, was die gesamte Entwicklung des *Trusted Computing* problematischer und fragwürdiger gestaltet. Zwar dürfte dies bei den meisten Heimanwendern keine ernsthaften Bedenken herbeirufen. Der Industrie sollte dieser Fakt allerdings Grund genug sein, TC nicht als Allheilmittel gegen Softwarepiraterie zu betrachten.

3.3.2 Digital Honesty

Die Behinderung und das Erschweren der Möglichkeit des Raubkopierens sind durch technische Mechanismen realisierbar. Einen anderen Ansatz zur Eindämmung von Softwarepiraterie steckt hinter dem Begriff *Digital Honesty*. Das Institut für Strategieentwicklung in Kooperation mit der Universität Witten/Herdecke und der Lehrstuhls für Soziologie der Fakultät für das Studium fundamentale der Universität Witten/Herdecke führten eine Studie durch, „[...]die empirische Daten aus Recherchen, Experteninterviews und einer Online-Umfrage systematisch nutzt.[...]“ Das Ziel war die Entwicklung eines Verständnisses, welche Faktoren Raubkopieren und die Nutzung von raubkopierter Software zu Grunde liegen und begünstigen. Darauf basierend sollten neu Konzepte zur Raubkopiebekämpfung erarbeitet werden.

Es stellte sich heraus, dass Softwarepiraterie als ein gesellschaftliches Phänomen angesehen werden kann, das Ausdruck menschlicher Ambivalenz ist. So ist Raubkopieren als Straftat anerkannt und ebenso, dass betroffene Unternehmen finanziellen Schaden erleiden. Andererseits folgt das eigene Handeln nicht dem Bewusstsein. Vor allem die Nutzung und Weitergabe raubkopierter Software gehört zum alltäglichen Leben. Die Studie kam zu dem Schluss, dass das bisherige Eigentumsverständnis nicht ausreicht, um innerhalb des geschaffenen virtuellen Raums zu greifen. Das Unrechtsbewusstsein der meisten Menschen ist nicht stark genug ausgeprägt, um einen Einfluss auf ihr Verhalten zu nehmen. Die Virtualität lässt den Tatbestand der Wegnahme und damit der Nachvollziehbarkeit einer Straftat vermissen. Erst wenn diese aus der realen Welt stammenden Erfahrungen in den virtuellen Raum eingebunden und stark genug in das Bewusstsein der Menschen getreten sind, können Verhaltensänderungen erreicht werden. Dies wird ein langwieriger Prozess sein. Der Vorteil für Konsumenten als auch für Industrie ist die Schaffung gegenseitigen Vertrauens und dem Gefühl, nicht als Feinde zu agieren. DRM kann dann unterstützend als Wahrung der Industrieinteressen eingesetzt werden.

Softwaredpiraterie

Die Studie differenziert vier verschiedene Nutzergruppen. Das ist sinnvoll, da einerseits das Verhalten von Raubkopierern und Raubkopienutzern nicht einheitlich erfasst werden kann. Andererseits können die Gruppen unterschiedlich angesprochen werden, um so die Möglichkeit eines Dialogs mit der Industrie zu eröffnen und eventuell neue Kundenverhältnisse zu erschließen.

Die Gruppen werden kategorisiert in PC-Freak, Hobby-User, Pragmatiker und PC-Profis.¹⁶

Die PC-Freaks sind versierte Nutzer, die mit hervorragenden Computerkenntnissen aufwarten können. Sie sind am ehesten in der Lage, die Mechanismen kopiergeschützter Software auszuhebeln. Ihr Besitz ist ein großes Reservoir von Raubkopien, mit denen sie auch arbeiten und die sie großzügig an ihnen bekannte Gruppen (Freunde, Verwandte, etc.) verteilen. Weiterhin nutzen sie die Anonymität innerhalb von Tauschbörsen aus, um die Verteilung „gecrackter“ Produkte voranzutreiben.

Der Hobby-User ist dem PC-Freak ähnlich. Er kopiert alles was er bekommen kann. Sein Wissen bzgl. Computer ist geringer als das des PC-Freaks. Bei Problemen wird nicht selbst recherchiert, vielmehr wird jede Hilfe angenommen.

Die Pragmatiker bezeichnet die größte Gruppe. Raubkopien existieren hauptsächlich dann, wenn sie auch verwendet werden. Ihr Wissen reicht nicht aus, um die Schutzmechanismen umgehen und/oder unwirksam machen zu können.

Die PC-Profis sind die gesetzeskonformsten Gesellschaftsmitglieder. Sie besitzen und nutzen kaum raubkopierte Software. Deren Besitz und Verwendung verstehen sie als klaren Rechtsverstoß. Sie verstehen und beherrschen ihre Computer bestens und nutzen ihn als Arbeitsinstrument.

Es mag vielleicht verwundern, dass die Gruppe der „Cracker“ nicht genannt wird, da sie durch ihre Kenntnisse den eigentlichen Schaden verursachen. „Cracken“ bezeichnet allgemein einen Angriff auf vorhandene Kopierschutzverfahren. Es ist zu vermuten, dass es für die Studie diese Einteilung nicht zweckmäßig gewesen ist. „Cracker“ sind in der vorhandenen Einteilung in allen Gruppen außer den Pragmatikern anzutreffen, wobei ein Großteil bei den PC-Freaks zu finden ist. Damit wäre eine Neueinteilung nötig gewesen, die wahrscheinlich mit den sozialen und gesellschaftlichen Einflussfaktoren in Widerspruch gestanden hätte.

Über alle Gruppen steht die Frage nach dem Verständnis von Eigentum. Die historische Prägung stellt den Käufer nach Erwerb einer realen Ware als Besitzer und Eigentümer dar.

¹⁶ Die Studie gibt neben allgemeinen Merkmalen auch den sozialen Stand und Altersdurchschnitt an. Hier wird dieser Teil explizit ausgespart, da der Autor der Meinung ist, dass dies die jeweiligen Gruppen denunziert und ihnen einen teilweise unrechtmäßigen Stempel aufdrückt

Softwarepiraterie

In einer digitalisierten Welt gibt es einen Bruch zwischen dem evtl. physischen Anteil und dem Inhalt einer Ware. Als Beispiel kann der Kauf eines Produkts auf einer CD-ROM herangezogen werden. Das Trägermedium, hier die CD-ROM, ist nach dem Kauf Eigentum des Käufers. Der auf dem Datenträger befindliche Inhalt ist es hingegen nicht. Vielmehr hat der Käufer eine für ihn persönliche Nutzungslizenz erworben, die ihn mit mehr oder minder eingeschränkten Verfügungsrechten ausstattet. So verhält es sich bei virtuellen Waren ebenfalls. Da der physische Teil entfällt, erwirbt ein Käufer neben dem eigentlichen Produkt nur Lizenzen. Genau dieser Unterschied zwischen erworbenen Eigentum und dem Erwerb eines Verfügungsrechts ist noch nicht Teil des Gedankenguts der meisten Menschen.

Zusammenfassend kann gesagt werden, dass das Konzept der *Digital Honesty* auf einem langfristigen Prozess gegenseitiger Vertrauensbildung der Industrie und Konsumenten basiert. Es werden zusätzlich zu technischen Möglichkeiten soziale, ethische und gesellschaftliche Gegebenheiten und Konventionen berücksichtigt. Konsumenten werden nicht „alle über einen Kamm geschert“, sondern in Gruppen differenziert wahrgenommen. Die Hauptaufgabe liegt in der Ausweitung des Verfügungs- und Eigentumsverständnisses auf den virtuellen Raum. Das Ziel ist Verständniserweiterung für den abstrakten Begriff des geistigen Eigentums. Folglich würde etwas wie ein digitales Selbstverständnis entstehen. Jeder Mensch soll sich seiner Handlungen bewusst sein, so als wenn sie in der realen Welt geschehen würden. Die Theorie stellt eine Verhaltensänderung in Aussicht, nach der weniger Konsumenten die Nutzung und Verbreitung raubkopierter digitaler Waren unterstützen. Konsumenten hingegen brauchen das Vertrauen in die Industrie, dass ihre persönlichen Angaben und Daten vertrauensvoll behandelt werden und sie für Qualitätsware nicht ausgebeutet werden. Der gegenseitige Respekt würde sich für alle Parteien in einem friedvolleren Miteinander und verbesserter Kommunikation ausdrücken.

3.4 Fair Play auf beiden Seiten

Die beiden vorangegangenen Kapitel informierten über verschiedene Möglichkeiten zur Eindämmung bzw. Behinderung von Softwarepiraterie mittels technischer Mechanismen und gesellschaftlicher Maßnahmen. Beide Konzepte haben ihre Vor- und Nachteile. Sie sollten trotzdem zusammen eingesetzt werden, um sowohl den Bedürfnissen der Industrie als auch der Konsumenten entgegen zu kommen. Softwareunternehmen sind auf die Rückführung ihrer Investitionen angewiesen. Bis ein gedanklicher Wandel innerhalb der Gesellschaft gegenüber geistigem Eigentum und dessen Diebstahl etabliert ist, müssen technische Schutzmechanismen als Argument zur Durchsetzung der Interessen dienen.

Softwairpiraterie

Es werden immer wieder neue Kopierschutzmechanismen und –Technologien entwickelt, und immer wird es einem professionellen Cracker gelingen, diesen Schutz zu umgehen und das Verfahren oder das Produkt selbst zu veröffentlichen bzw. zu verteilen. Die Androhung drakonischer Strafen seitens der Industrie hat bisher keinen abschreckenden Charakter gehabt. Stattdessen sind neue Ansätze gefragt, die beide Seiten fair behandeln. Solch einen Ansatz geht das PotatoSystem (PS) nach, das im folgenden Kapitel 4 beschrieben wird.

4 Das PotatoSystem

„[...]Das PotatoSystem ist eine zentrale Dienstleistung zum dezentralen Vertrieb virtueller Güter für Erzeuger und Nachfrager. Es ist für alle gängigen Medienformate geeignet zum Beispiel für MP3-Dateien. [...]“⁽¹³⁾

4.1 Motivation

Die Hauptintention dieses Projekts war die Schaffung eines Systems, das die Interessen von Konsumenten und Wirtschaft gleichermaßen durch einen fairen Kompromiss wahren soll. Das zugrunde liegende Vermarktungsmodell soll Anreize für Konsumenten schaffen, virtuelle Waren zu erwerben und deren Verbreitung zu fördern. Es wird davon ausgegangen, dass Konsumenten ihre Freunde, Verwandtschaft, Arbeitskollegen und sonstige Bekanntschaft mit ähnlichem Interesse auf Waren aufmerksam machen. Die Güter würden dadurch bei immer mehr Personen bekannter werden. Dieses Prinzip wird als *Super Distribution* über *Super Distribution Channels* bezeichnet. Der Kauf einer Ware schließt den Erwerb des Wiederverkaufsrechts ein. Der Käufer kann dadurch selbst als Vertriebspartner tätig werden. Das vorhandene Provisionsmodell regelt die Verteilung der Verkaufserlöse. Somit ist eine faire Behandlung aller Beteiligten theoretisch möglich. Anonyme Weitergabe kann natürlich nicht komplett ausgeschlossen werden. So wird im Falle des Anbietens von Musik bewusst auf einen Kopierschutz verzichtet, um Nutzern eine hohe Nutzungsfreiheit zu garantieren. Dadurch kann auf beiden Seiten ein Vertrauensverhältnis aufgebaut werden, das einen fairen und respektvollen Umgang begünstigt.

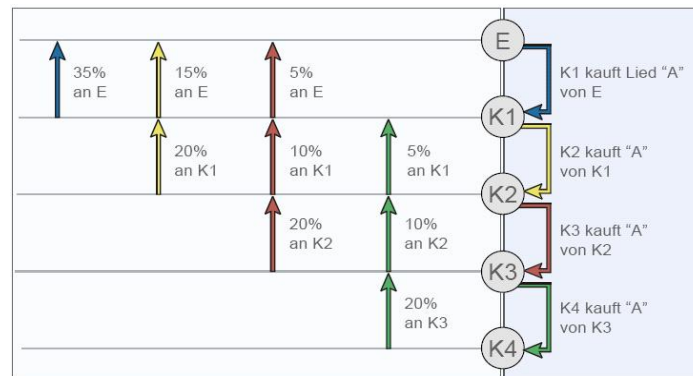
Das PS sollte nicht ausschließlich ein Verteilungssystem virtueller Waren für professionelle Verkaufsportale, Labels jedweder Größenordnung und Selbstvermarktungskünstlern sein. Vielmehr geht es um die Schaffung eines Gemeinschaftsgefühls zwischen Konsument und Anbietern. Die Vorstellung, sich Gleichgesinnten anzuschließen, die ebenfalls auf die Verbreitung qualitativ hochwertiger virtueller Waren ohne Restriktionsmechanismen Wert legen, soll den fairen Umgang miteinander weiter begünstigen.

4.2 Geschäftsmodell und Dienstleistungen

Das Geschäftsmodell des PotatoSystems kann nach (9) als hybrides Geschäftsmodell bezeichnet werden. Es ist in erster Linie auf die transaktionale Abwicklung von Downloads bezahlter digitaler Waren ausgelegt (Download-on-Demand) und fungiert als Vermittler zwischen Produzenten/Anbietern und Käufern/Konsumenten. Das Kernstück bildet das Provisionsmodell.

Das PotatoSystem

Hierbei handelt es sich um ein als Multilevel-Affiliate-Marketingprogramm bezeichnetes Vertriebskonzept. Die Verkaufsstruktur wird dabei hierarchisch gegliedert, mit dem Ziel, die Konsumenten einzuspannen und Anteil am Verkauf zu haben.



Legende: E = Erstverkäufer (z.B. Label), K1 ... K4 = Käufer 1 bis 4

Abbildung 2 Provisionsmodell des PotatoSystems (entnommen aus (10))

Anbieter erhalten im für sie ungünstigsten Fall nur noch 30% des ursprünglich festgelegten Warenpreises, im günstigsten Fall 85%. Dies ist zum einen abhängig vom verwendeten Bezahlssystem, welches einen festgelegten Anteil für den angebotenen Dienst einbehält, was zwischen 10% bis 30% ausmachen kann. Zusätzlich gehen 5% für das PS ab. Zum anderen ist entscheidend, ob eine Ware direkt oder über einen Weiterverkauf durch einen Konsumenten verkauft wird. Das klingt zunächst wie ein unrentables Geschäft. Doch der Versuch, die Konsumenten in den Verkaufsprozess zu integrieren kann diesen Umstand durch vermehrte legale Warenkäufe zu einer positiven Bilanz drehen. Weiterverkäufer können bis zu 35% des festgelegten Preises einer Ware durch Wiederverkauf bekommen.

Zusätzlich kann beim Verkauf von Musiktiteln ein weiterer Anteil an die „Gesellschaft für musikalische Aufführungs- und mechanische Vervielfältigungsrechte“ (GEMA) entfallen. Dies ist aber nur dann der Fall, wenn der Produzent einen Vertrag mit der GEMA hat oder das PS Anwendung in der deutschen Musikbranche findet.

Es werden verschiedenste Dienstleistungen angeboten, um den unterschiedlichen Nutzerbedürfnissen entgegen zu kommen. Alles basiert auf dem Einfachheitsprinzip, um eine komfortable und unkomplizierte Bedienung des Systems für die Nutzer zu ermöglichen.

Nutzer können allgemein in die Rollen Konsumenten und Anbieter unterschieden werden. Letztere registrieren sich und ihre zu verkaufenden Waren beim PotatoSystem.

Das PotatoSystem

Sie können einen aus drei verschiedenen Anbieter-Accounttypen¹⁷ wählen, die entsprechende Vergütung für das PS mit sich bringen. Je nach Wahl erhält ein Anbieter verschiedene Beschränkungen und Unterstützungen seitens des Systems. Um nicht jede Ware einzeln und manuell registrieren zu müssen, existiert für die Accounttypen Label und Portal ein komfortabler Mechanismus zur Massenregistrierung. Dieser kann ebenfalls genutzt werden, um Daten für die Offline-Verwaltung der Produkte zu erhalten, z.B. Verkaufsstatistiken.

Mit der Registrierung ihrer Ware, willigen Anbieter ein, dass beim Kauf der zahlende Konsumenten die Weiterverkaufsrechte und somit einen Provisionsanspruch erhält. Dies gilt sowohl für bereits registrierte Käufer, als auch für den anonymen Erwerb. Letzteres wird durch einen so genannten Aktivierungscode ermöglicht, mittels dessen ein anonymer Käufer nachträglich den Erwerb im PS registrieren kann. Voraussetzung dafür ist die erfolgreiche Anmeldung eines Nutzer-Accounts. Jeder Anbieter kann auch die Rolle eines Käufers einnehmen und wird somit ebenfalls Wiederverkäufer.

Das Informationsangebot des PotatoSystems ist recht umfangreich. Die Konsumenten erhalten Informationen zu sämtlichen registrierten Waren. So werden bspw. zu einzelnen Musikstücken der Preis, Künstler, Albumzugehörigkeit, etc. angegeben. Jeder registrierte Nutzer erhält, nachdem er sich eingeloggt hat, eine Übersicht über alle bisherigen Transaktionen. Darunter fallen die selbst erworbenen und weiterverkauften Waren. Ebenfalls ist ein Webspace für eine persönliche Website des jeweiligen Konsumenten reserviert, um auch Weiterverkäufe für diejenigen zu ermöglichen, die keinen eigenen Webauftritt besitzen.

Trifft ein potentieller Käufer eine Auswahl von Waren, die er eventuell erwerben will, werden diese zuerst in einen persönlichen Warenkorb gelegt. Die darin enthaltene Auswahl kann jederzeit modifiziert werden. Nach dem Kauf von einer oder mehreren Ware/n wird eine Liste anderer Kunden angezeigt, die ebenfalls diese/n Artikel erworben haben. Dazu wurde ein Algorithmus namens *Potato-Match* (siehe (16)) entwickelt. Die Konsumenten können in Kontakt miteinander treten und sich gegenseitig auf Waren aufmerksam machen oder voneinander kaufen. Dadurch soll der „Community“-Charakter durch das System unterstützt werden.

4.3 Architektur

Das PotatoSystem ist als eine Art Baukastensystem konzipiert, das als Backend-Lösung für kleine und größere Download-Portale genutzt werden kann.

¹⁷ Basic-, Label- und Portal-Account

Das PotatoSystem

Die folgenden Abbildung 3, als FMC Block Diagram¹⁸ modelliert, zeigt auf hoher Abstraktionsebene die grundlegende Architektur des PotatoSystem und seine Umgebung. Die Farbgebung ist willkürlich und nur zur besseren Unterscheidung der verschiedenen Modellierungselemente eingeführt.

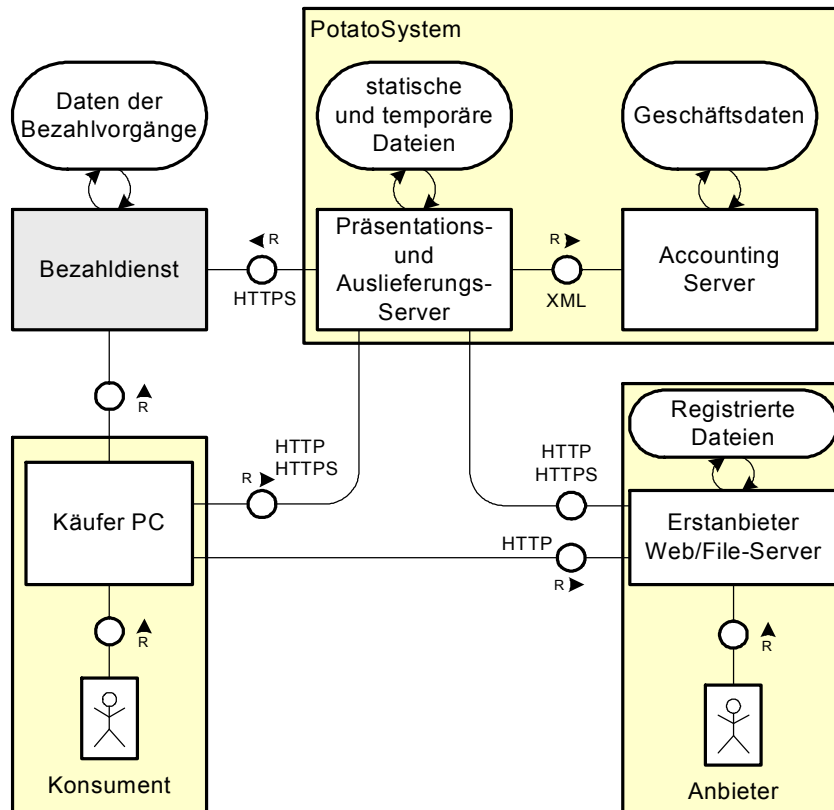


Abbildung 3 Einbettung des PotatoSystems in seine Umgebung

Das PotatoSystem (PS) basiert auf zwei Teilsystemen, dem Accounting Server (AS) und einem Präsentations- und Verbreitungs-Server. Das Herzstück bildet der AS, der das Backend-System des PS bildet. Hier werden sämtliche relevante Daten verwaltet und wichtige Dienste angeboten. Jede registrierte Datei erhält eine eindeutige Identifizierungsnummer mit Verweis auf ihre Lokalität. Es werden alle im PS stattfindenden Transaktionen ausgeführt und gespeichert. Die Nutzeraccounts werden verwaltet und die Provisionen bei Kauf einer registrierten Ware berechnet. Der AS bietet keine direkte Kommunikationsschnittstelle für die Umgebung des PS an. Dafür ist das zweite Teilsystem zuständig. Es handelt sich um mehrere Server, welche die Dienste und Angebote des PS mittels graphischer Nutzeroberfläche anbieten. Der Datenaustausch mit dem AS erfolgt über ein XML-basiertes Kommunikations-Protokoll.

¹⁸ FMC – Fundamental Modeling Concepts – ist eine Modellierungssprache zur Beschreibung von software-intensiven System. Ihr Einsatzgebiet ist die Modellierung logischer Systemstrukturen und –Abläufe, sowie die Identifizierung und Lokalisierung der relevanten Werte und Wertestrukturen. Detaillierter Informationen, siehe (25)

Das PotatoSystem

Alle Eingaben und Dienstaufrufe sämtlicher Nutzer, ob registriert oder anonym, werden entgegengenommen und entsprechend behandelt. Zum Teil werden Dateien zwischengespeichert, um einerseits die Skalierbarkeit des Systems zu erhöhen, andererseits performantere Antworten auf eingehenden Downloadanfragen zu ermöglichen. Die Kommunikation läuft über HTTP oder HTTPS. Letzteres bei sicherheitskritischen Eingaben, beispielsweise Login oder Kauf einer Ware. Ebenfalls über HTTPS läuft der Datentransfer mit angebotenen Bezahldiensten. Die 4FO AG betreibt ein eigenes Multi-Payment-Broker System, welches Zugang zu einem selbst entwickelten Bezahlssystem als auch Drittanbietersystemen wie PayPal oder FirtsGate bietet.

Die folgende Abbildung 4 zeigt die logische Architektur des PS. Hierbei wurde die Kommunikation mit den Bezahldiensten außer Acht gelassen. Schwerpunkt bildet die Beschreibung des PS und seiner internen Systemteile.

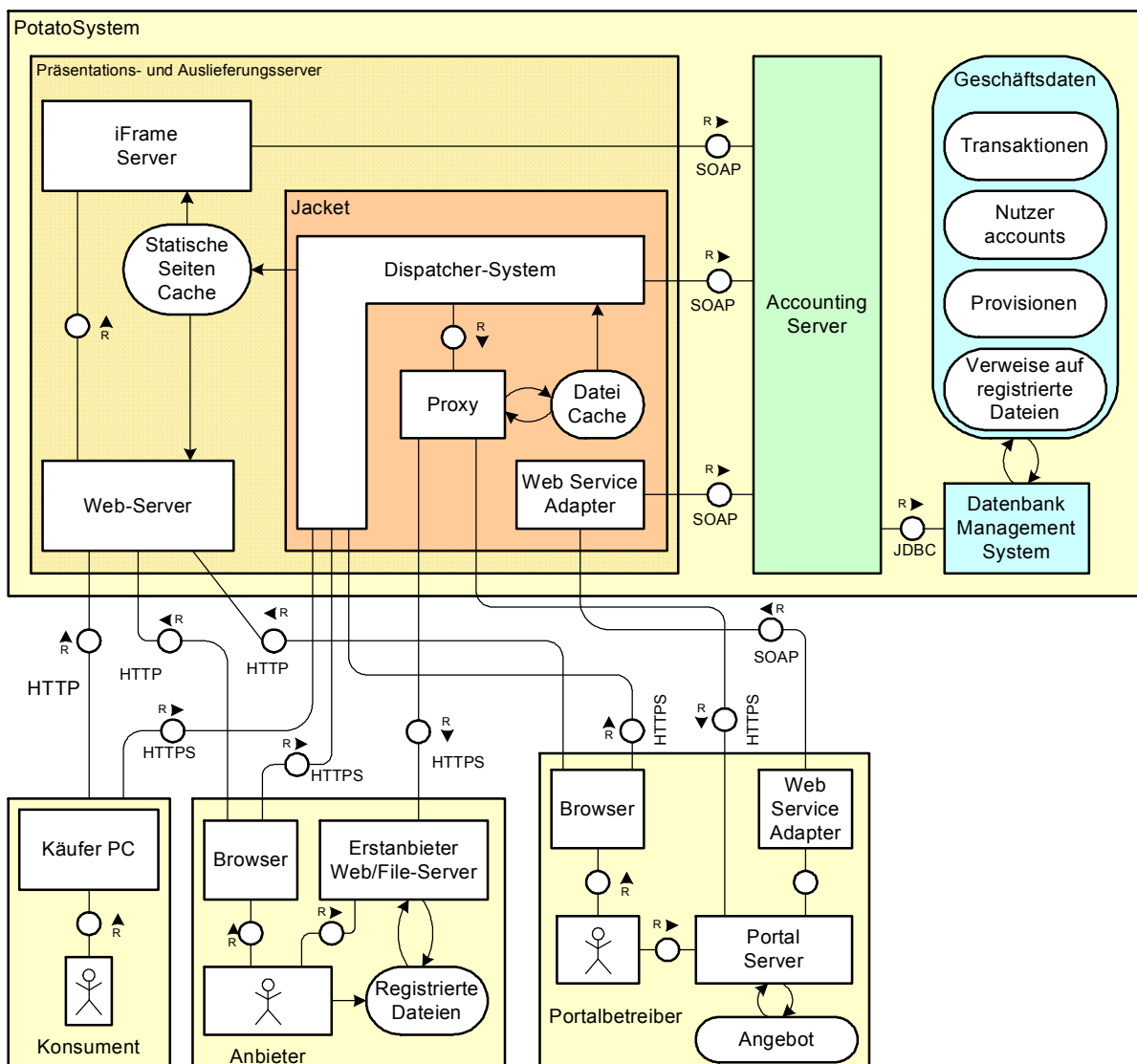


Abbildung 4 Architektur des PotatoSystems

Das PotatoSystem

Der Präsentations- und Auslieferungsservers kann in drei Subsysteme unterteilt werden. Der Web-Server nimmt alle Client-Anfragen die über HTTP an das System gestellt werden entgegen. Dabei handelt es sich grundsätzlich um einen unkritischen Datenaustausch, bspw. die Übermittlung der Startseite. Erst bei sicherheitsrelevanten Anfragen wird der Kommunikationskanal gewechselt. Dann geht die Kontrolle auf das so genannte Jacket über. Das Jacket besteht selbst wiederum aus mehreren Teilsystemen. Für die Steuerung, Verarbeitung von Anfragen und Auslieferung von Dateien ist ein spezielles Dispatcher-System zuständig. Alle Datentransfers beruhen auf HTTPS, z.B. das UserLogin oder die Auslieferung virtueller Waren. Die Komponenten, die als Proxy beschrieben wird, hat zwei Aufgaben zu erfüllen und ist technisch in zwei verschiedene Kategorien einzuteilen. Zum einen wird er genutzt, um Dateien, die nicht im internen Cache liegen, vom Anbieterserver zu holen und im Cache abzulegen. Somit können schnellere Antwortzeiten auf Download-Anfragen ermöglicht, sowie die Skalierbarkeit des Systems erhöht werden. Er nimmt hier die Rolle eines Cache-Managers ein. Gleichzeitig kann er als Stellvertreter für Anbieter auftreten, deren Produkte häufig angefragt werden und bereits im Cache liegen. Somit erfüllt er in dieser Rolle die in (53) beschriebenen Eigenschaften, die ihn zu einem wirklichen Proxy machen.

Eine Besonderheit ist die angebotene Web-Service-Schnittstelle für Portalbetreiber. Portale sind komplexe Systeme, die das PS als *Application Service Provider* (ASP) nutzen können. Sie können hierüber verschiedene Metadaten ins PS einpflanzen oder herausziehen, als auch Verkaufsstatistiken erhalten, die jederzeit abrufbar sind.

Der iFrame-Server ist speziell für die Auslieferung der Weiterverkaufslinks an die Käufer gedacht. Für jede erfolgreiche Transaktion wird eine systemweit eindeutige Transaktionsnummer (TAN¹⁹) vom Accounting-Server generiert, die in den Weiterverkaufslink integriert wird. Somit können die bei einem Kauf beteiligten Parteien eindeutig identifiziert werden. Der für jede Datei erhaltene Weiterverkaufslink kann dann vom Käufer veröffentlicht werden oder der Käufer kann einen eigenen Link erschaffen. Dabei müssen die speziellen Daten exakt übernommen werden, da es sonst zu Konflikten kommen kann und Fehlverhalten bei (Weiter)Verkäufen die Folge wäre.

4.4 Abläufe

Die Abläufe sollen für die Konsumenten als auch für die Anbieter so einfach wie möglich gehalten werden. Das PS nimmt die Rolle eines Vermittlers ein. Die zum Verkauf angebotenen Waren müssen von den Erstanbietern registriert und die Eingaben vom System validiert

¹⁹ Der Aufbau einer TAN wird in (11) beschrieben

Das PotatoSystem

werden. Im Falle von Musikdateien sind die Anbieter verpflichtet, die ID3-Tags korrekt zu bearbeiten, eine kostenlose Vorhördatei zur Verfügung zu stellen und die zum Verkauf bereitgestellte Ware innerhalb eines geschützten Verzeichnisses zu platzieren, auf welches nur das PS Zugriff hat. Konsumenten können diese Artikel erwerben und evtl. weiterverkaufen. Sowohl bei der Registrierung als auch bei jedem Kauf wird eine systemweit eindeutige Transaktionsnummer (TAN) erzeugt und im Backend-System gespeichert. Sie wird im Dateinamen einer legal erworbenen Kopie integriert und dient als Kaufbeleg für den Käufer. Damit dieser den Artikel weiterverkaufen kann, erhält er einen Verkaufslink, in dem ebenfalls die TAN eingefügt wurde. Es werden die nötigen Provisionen berechnet und gutgeschrieben. Zusätzlich besteht die Möglichkeit, anonyme Transaktionen nachträglich anzumelden, um nicht registrierten Käufern die Weiterverkaufsrechte zu übertragen. Konsumenten können einmal bezahlte Waren jederzeit erneut anfordern und herunterladen.

Die folgende Abbildung 5 beschreibt den allgemeinen Ablauf eines einfachen Kaufvorgangs, bei dem sich der Käufer im System registriert und angemeldet hat.

Der grundlegende Kaufvorgang bleibt prinzipiell bei jedem Erwerb virtueller Waren bestehen. Es existieren weitere Varianten für Kaufvorgänge wie der anonyme Kauf mittels Aktivierungscodes. Es wird hier nicht weiter darauf eingegangen, da nur ein grober Überblick gegeben werden soll. Ausführlichere Beschreibungen und Verweise auf Quellen sind bei (1) und (2) zu finden.

Das PotatoSystem

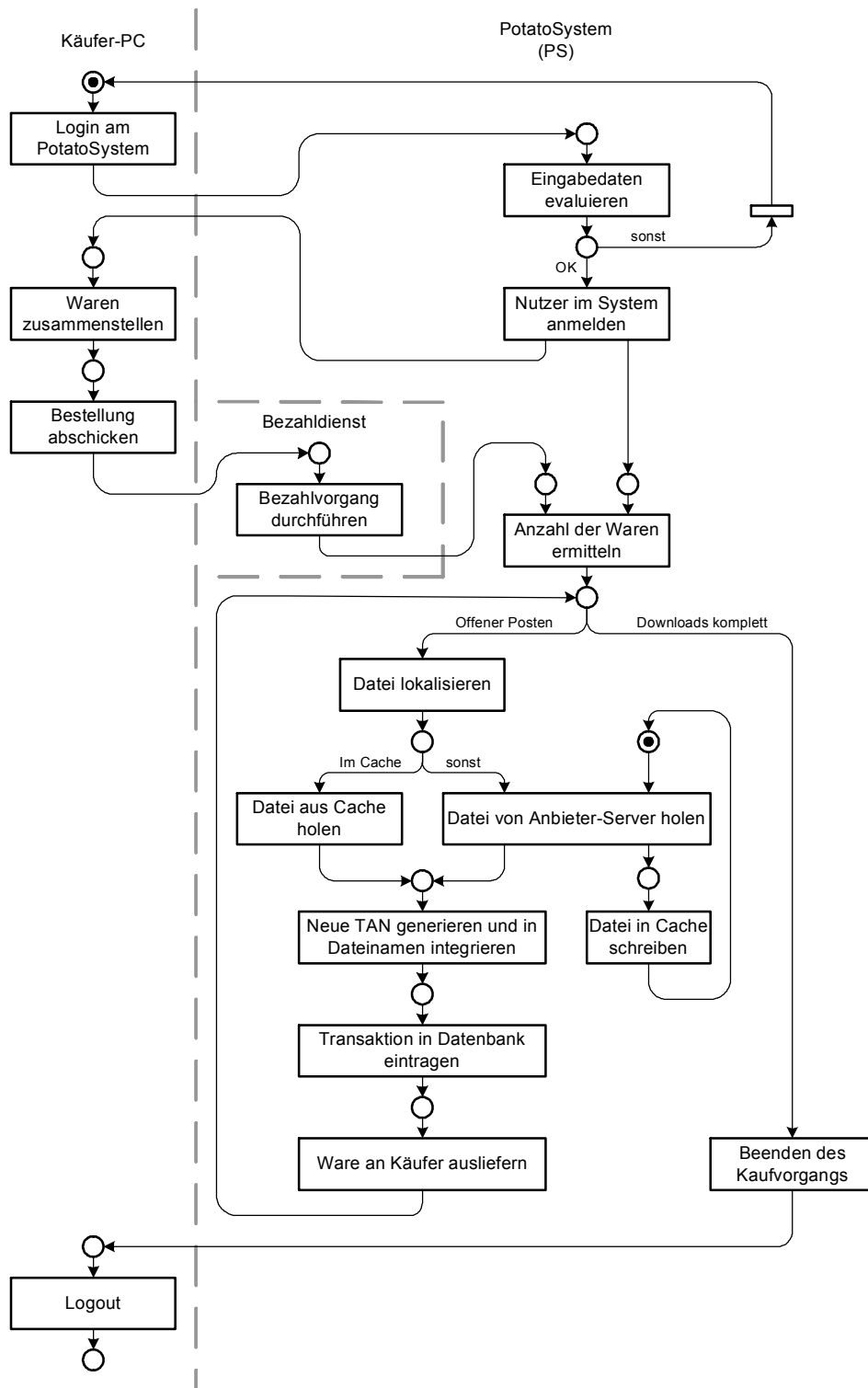


Abbildung 5 einfacher Kaufvorgang

5 Konzeption einer Erweiterung zur elektronischen Software-Distribution

Nachdem das PotatoSystem (PS) in seinen Grundzügen vorgestellt wurde, sollen jetzt verschiedene Konzepte zur Erweiterung betrachtet und auf ihre Vor- und Nachteile untersucht werden. Nachdem die allgemeinen Anforderungen beschrieben wurden, sind die Unterschiede zwischen elektronischer Musik- und Software-Distribution herauszustellen. Darauf aufbauend können dann verschiedenste Ansätze für die Verwirklichung einer Erweiterung des PS zur elektronischen Software-Distribution (ESD) erarbeitet und auf ihre Zweckmäßigkeit und Machbarkeit bewertet werden.

5.1 Anforderungen an die ESD-Erweiterung

Die zu entwickelnde Erweiterung soll nach folgenden Anforderungen und Vorgaben durch die 4FO AG konzipiert sein. Sie soll

- minimal-invasiv sein, um bestehende Systemstrukturen und deren Abläufe nicht oder nur im geringst möglichen Maße zu erweitern bzw. zu modifizieren. Z.B. sollen wie in der elektronischen Musik-Distribution die Anbieter ein öffentliches und ein geschütztes (Content)Verzeichnis auf ihren Servern erstellen
- dem KISS-Prinzip (*Keep it stupid and simple*) folgen, um die Integration ins bestehende System zu vereinfachen und den damit verbundenen Aufwand niedrig zu halten
- für alle Nutzer einfach zu bedienen sein. Das bedeutet nicht, dass die Nutzer für unfähig gehalten werden, komplexe Anforderungen zu begreifen. Vielmehr soll die Benutzbarkeit einfach und effektiv gestaltet sein.

5.2 Unterschiede zur Musik-Distribution

Generell können Musikdateien als passive Entitäten bezeichnet werden. Sie führen sich nicht selbst aus, sondern werden durch ein anderes Programm verarbeitet, welches die enthaltenen Daten und Metainformationen interpretiert. Neben dem eigentlichen Inhalt existieren verschieden Metadaten, z.B. über den Künstler oder auch Lizenzinformationen für DRM-Systeme. Softwareprogramme hingegen sind aktiv, in dem Sinne, dass sie selbstständig ausgeführt werden können und nur durch spezielle Technologien oder Plattformen beschränkt sind. Evtl. Schutzmechanismen sind in einem Programm integriert. Die zum Ablauf benötigten Daten werden durch die Software selbst evaluiert.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Das PS wird zurzeit überwiegend für den Vertrieb von Musikdateien verwendet. Die Werke werden innerhalb eines geschützten Verzeichnisses gehalten, auf das nur das PS Zugriff hat. In einem weiteren, öffentlichen Verzeichnis existieren so genannte *Previews*, die einem Konsumenten einen ersten Einblick in das geschaffene Werk erlauben. Diese sind im Allgemeinen zeitlich auf etwa 45 Sekunden beschränkt und von minderer Qualität als das zu verkaufende Musikstück. Wird ein Werk erworben, so wird in den Dateinamen die TAN als Kaufbeleg eingefügt. Zudem darf der Käufer die Datei so oft es ihm beliebt herunterladen. Die Verfügbarkeit von Dateien wird durch Zwischenspeicherung im Proxy-Cache des Jacket-Servers erhöht. Das PS erlaubt darüber hinaus den anonymen Kauf von Musiktiteln mittels so genannter Aktivierungscodes.

Die vorhandenen Strukturen und Abläufen sollen für die Verteilung von Software weitestgehend übernommen werden, so z.B. die Existenz eines öffentlichen und geschützten Bereichs auf dem Anbieterserver. Öffentlich können die Programme sein, die den Käufern als Testversion zur Verfügung gestellt werden sollen. Der geschützte Bereich enthält dann je nach Konzept Registrierungsgeheimnisse und/oder weitere Teile der zu erwerbenden Waren.

Das verwendete Quittierungsverfahren, bei dem eine TAN in den Dateinamen eingefügt wird, kann bei Softwareprogrammen unter Umständen problematisch sein. Im schlimmsten Fall kann die ausgelieferte Kopie nicht mehr gestartet werden. Das ist der Fall sein, wenn die Integritätsprüfung der Datei durch einen Kopierschutzmechanismus fehlschlägt. Zudem muss eine Software nur einmal herunter geladen werden, für die dann mehrere Produktschlüssel erworben werden können. Daher wird die Quittierungsmethode über TANs als unzuverlässig angesehen.

Die Anbieter werden schon heute in die Pflicht genommen, ihre Waren immer verfügbar auf einem Web/File-Server zu halten. Daher werden unabhängig vom gewählten ESD-Ansatz auszuliefernde Softwareprogramme nicht im internen Proxy-Cache zwischengespeichert. Dies hat aus PS-Sicht den Vorteil, dass der File-Cache nicht überladen wird. Es ist durchaus möglich, dass ein Programm mindestens doppelt so groß wie eine durchschnittliche Musikdatei ist. Dies kann der Fall sein, wenn alle zu einem Programm gehörenden Artefakte in eine einzige ausführbare Datei kompiliert werden. Ein Beispiel sind anbieterspezifische, eingebundene Bibliotheken, die das Programm stark „aufblähen“ können.

Ein weiterer kritischer Punkt ist die anonyme Auslieferung eines Produkts. Ein Nutzer könnte sich viele Kopien eines Programms herunterladen und dann versuchen, den vorhandenen Kopierschutz- bzw. Sharewaremechanismus zu brechen.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Dadurch erhielte dieser Nutzer die Möglichkeit, einen Key-Generator zu erstellen und zu veröffentlichen bzw. Listen mit gültigen Produktschlüsseln online verfügbar zu machen. Das gleiche gilt für den anonymen Kauf von Produktschlüsseln. Besitzt ein einzelner Nutzer viele Schlüssel eines Produkts, so kann er ebenfalls versuchen, Rückschlüsse auf den Algorithmus zur Evaluierung des Schlüssels zu ziehen. Findet er den Mechanismus heraus, so kann er ebenfalls gültige Listen und/oder Key-Generatoren veröffentlichen. Daher sollten das Herunterladen einer Kopie sowie der Kauf von Produktschlüsseln nur im PS registrierten Nutzern gestattet werden. Somit kann teilweise Missbrauch ausgeschlossen werden. Ebenfalls wäre es möglich, veröffentlichte Produktschlüssel zu überprüfen, ob und wenn ja, von wem sie erworben wurden. Findet sich ein bereits erworbener Schlüssel auf einer veröffentlichten Liste wieder, sollte der Anbieter als auch Konsument über diesen Missstand informiert werden. Dieser Tatbestand reicht natürlich nicht aus, um festzustellen, ob der Konsument mit dem übereinstimmenden Produktschlüssel auch derjenige ist, der evtl. die Liste veröffentlicht hat. Der Schlüssel kann auch Ergebnis eines Key-Generators sein. Zumindest sollten Anbieter von ihrer Seite her versuchen, einen Dialog mit dem/den betroffenen Konsumenten aufzubauen, falls möglich.

5.3 Kategorisierung der Verteilungsstrategien

Die vorhandenen Strukturen und Abläufe des PS dienen als Grundlage für die Überlegungen zur Verteilung von Softwareprogrammen. Allgemein können zwei mögliche Kategorien unterschieden werden, die in den folgenden Kapiteln beschrieben werden. Beide Varianten zielen in Verbindung mit dem Geschäftsmodell des PS auf die Verbreitung von Produkten über *Super-Distribution-Channels*.

5.3.1 Auslieferung der Software als „Try-and-Buy“-Version

Die zu verkaufende Software wird als zeit- und/oder startbeschränkte Shareware-Version im öffentlichen Verzeichnis des Anbieters zum Download bereitgestellt. Hierbei handelt es sich um die Testversion des Programms, welche der Vorhördatei bei der Musikdistribution entspricht. Das PS liefert die Datei als Vermittler an den anfragenden Kunden aus, ohne eine Zwischenspeicherung vorzunehmen. Der interessierte Konsument muss im PS registriert und zugleich eingeloggt sein, damit ein Download stattfindet. Ein auf Käuferseite vollständiges ESD-System würde gleichzeitig die Installationsroutine des Programms anstoßen. Dem Konsumenten sollte diese Entscheidung jedoch freigestellt bleiben, bspw. wenn es sich um zeitbeschränkte Sharewareprodukte handelt, ist die sofortige Installation nicht immer erwünscht.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Der Konsument möchte vielleicht mehrere Programme einer bestimmten Kategorie ausgiebig testen. Da kann es hinderlich sein, wenn das Auslieferungssystem die Installation veranlasst und die benötigte Zeit zum Testen nicht mehr vorhanden ist.

Entscheidet der Konsument sich zum Kauf des Produkts, muss er einen Produktschlüssel erwerben, der die bisherige Testversion als Vollversion freischaltet. Die folgende Abbildung 6 zeigt den einfachen allgemeinen Ablauf eines Produktschlüsselerwerbs.

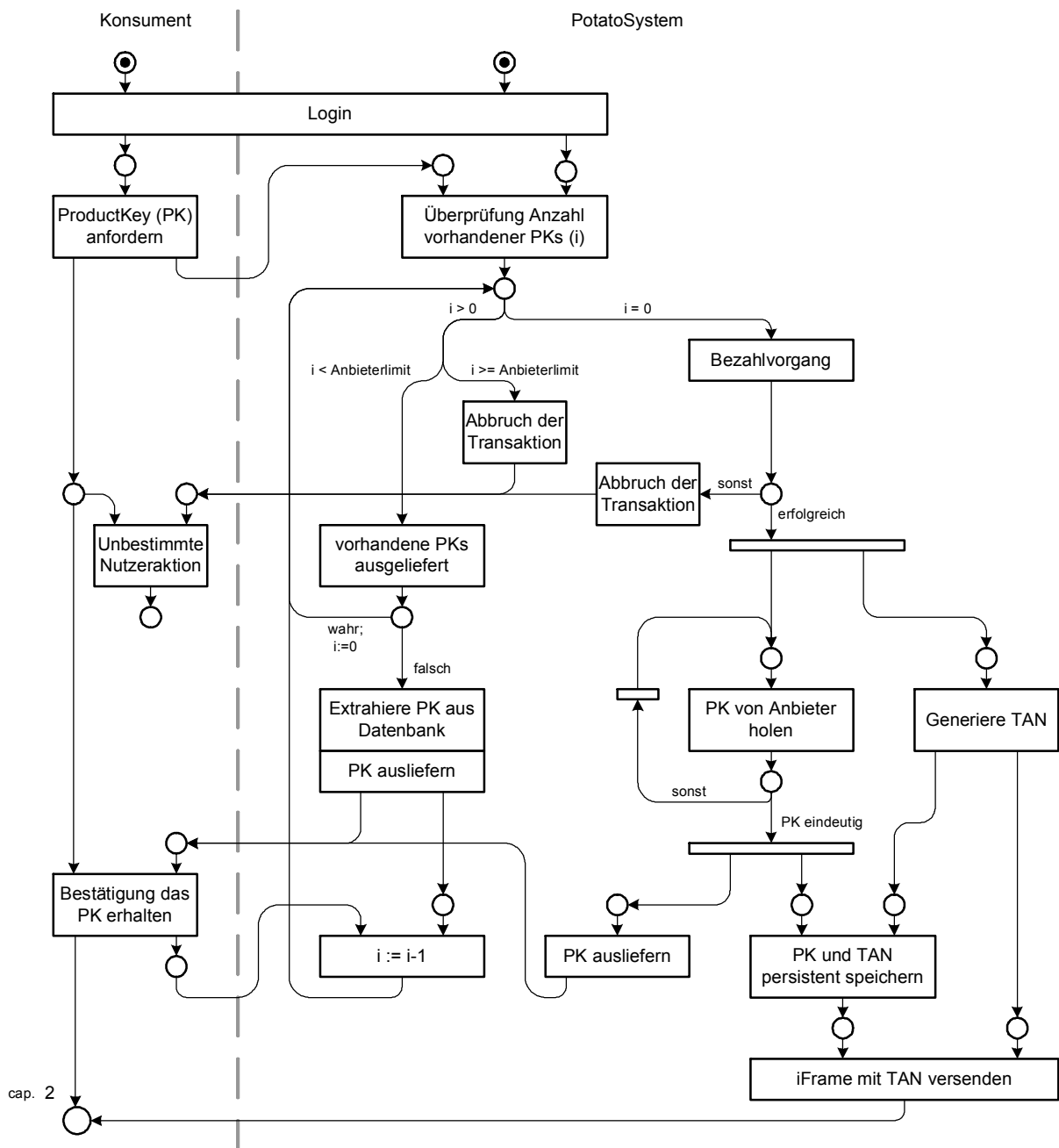


Abbildung 6 Kauf eines Produktschlüssels

Aus Gründen der Einfachheit soll angenommen werden, dass es sich um einen Produktschlüssel für eine Einzelplatzlizenz handelt.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Die Anfrage richtet sich an das PS, das zunächst überprüft, ob noch Schlüssel ausgeliefert werden dürfen. Dies kann unter Umständen nicht der Fall sein, wenn der Anbieter ein Limit für jeden Konsumenten gesetzt hat. Bei Überschreitung hat dies einen automatischen Abbruch der Transaktion durch das PS zur Folge. Der Käufer erhält dann den Ratschlag, sich mit dem Anbieter in Verbindung setzen. Hierbei handelt es sich um eine optionale Konfiguration. Die meisten Anbieter werden voraussichtlich keine Schranke wünschen. Falls der Wunsch doch besteht, sollte die Angabe optimalerweise bei der Registrierung des Produkts geschehen. Um Anbietern einen größtmöglichen Entscheidungsfreiraum zu lassen, kann dies später nachgeholt bzw. der bestehende Wert modifiziert werden. Darf der Konsument noch Produktschlüssel erwerben, wird überprüft, ob noch nicht ausgelieferte Schlüssel vorhanden sind. Falls dies der Fall sein sollte, wird versucht, die noch im PS verbliebenen legal erworbenen Schlüssel auszuliefern. Diese werden ebenfalls auf der persönlichen Transaktionsseite des Konsumenten einzeln angegeben und können einzeln herunter geladen bzw. kopiert werden.

Sind alle Bedingungen zum Erwerb auf Seiten des Konsumenten erfüllt, wird die Anfrage an den Anbieter weitergeleitet. Die Antwort ist ein Schlüssel, der vom PS auf seine Einzigartigkeit geprüft wird. Jeder Schlüssel kann einem Programm zugeordnet werden, weshalb durchaus zwei identische Produktschlüssel innerhalb des PS existieren können. Da sie aber verschiedenen Produkten und evtl. verschiedenen Anbietern zugeordnet werden können, würde der Test nur negativ ausfallen, wenn Produkt, Produktschlüssel und Anbieter identisch sind. Sollte dies der Fall sein, wird der Anbieter ein weiteres Mal mit der Anfrage kontaktiert. Kann der Schlüssel als einzigartig bestätigt werden, erfolgt die Auslieferung an den Konsumenten. Dabei gibt es verschiedene Möglichkeiten:

- der Produktschlüssel wird dem Konsumenten auf seiner persönlichen PS-Webseite angezeigt.
- der Produktschlüssel wird in eine Datei auf dem Computersystem des Konsumenten geschrieben, deren Speicherort dem Konsumenten mitgeteilt wird
- der Konsument erhält eine signierte E-Mail mit verschlüsseltem Inhalt

Die Auslieferung in den ersten beiden Möglichkeiten geschieht grundsätzlich über einen sicheren Kanal (siehe Abbildung 4). Die Anzeige über die vom PS bereitgestellte persönliche Webseite eines Konsumenten hat den Vorteil, dass sich der Käufer den Freischaltcode nirgends aufschreiben oder in einer Datei auf seinem Rechnersystem speichern muss. Er kann somit auch nicht verloren gehen, falls der Konsument sein Computersystem erneuern sollte, z.B. aufgrund einer zerstörten Festplatte, deren Datensätze nicht mehr lesbar sind. Zwar ist genau dies die Schwachstelle der zweiten Möglichkeit, aber der Konsument braucht nicht

Konzeption einer Erweiterung zur elektronischen Software-Distribution

online zu sein, falls er die Installation zu einem späteren Zeitpunkt durchführen will. Allerdings hieße dies, dass der Käufer dem PS erlauben muss, Dateien per Remote-Operation zu erstellen und zu modifizieren. Das bedeutet ein erhöhtes Sicherheitsrisiko für das Konsumentensystem und ist daher nur als schlechte Alternative anzusehen. Beide Varianten sind eine Dienstleistung für Kunden, denn diese brauchen sich nur noch um die Eingabe des übermittelten Freischaltcodes zu kümmern. Dies muss manuell geschehen, da die spezifischen Vorgänge des verwendeten Sharewaremechanismus dem PS nicht bekannt sind. Somit ist ein automatisches Einfügen nicht möglich.

Die Übermittlung per E-Mail ist nur dann eine Alternative, falls der Empfänger ebenfalls ein sicheres Programm zum Empfang besitzt. Als Beispiele sind PGP (*Pretty Good Privacy*) und seine freie Variante GnuPG (*Gnu Privacy Guard*) anzuführen. Allerdings kann eine E-Mail verloren gehen, sei es auf dem Weg zum Käufer oder der Käufer verliert sie selbst. Dann gäbe es keine Möglichkeit mehr für den Konsumenten, den bezahlten Produktschlüssel wieder zu bekommen. Daher muss als Absicherung mindestens einer der beiden vorangestellten Vorschläge realisiert werden.

Bei einer erfolgreichen Transaktion wird der verwendete Produktschlüssel innerhalb des PS gespeichert. Diese Maßnahme ist nötig, um einerseits die Einzigartigkeit des Schlüssels durch das PS überprüfen lassen zu können und andererseits, um ihn dem Konsumenten auf seiner persönlichen Transaktionsseite anzeigen zu können. Die generierte TAN wird dem Käufer als Kaufbeleg in einem iFrame integriert nach bestehenden PS-Abläufen zur Verfügung gestellt.

Der Anbieter wird bei dieser Variante bei jeder Kaufanfrage kontaktiert. Er ist dafür verantwortlich, dass ein gültiger Produktschlüssel zurück geliefert wird. Es kann allerdings passieren, dass ein Anbieter nicht verfügbar ist, z.B. durch einen Serverfehler, eine physische Leitung wurde durchtrennt, usw. Im einfachsten Fall ist dann das Herunterladen als auch der Kauf eines Produkts dieses Anbieters für diesen Zeitraum nicht möglich. Dies ist sowohl für Konsumenten als auch für Anbieter ein Ärgernis und äußerst unpraktikabel. Da das PS keine Softwaredateien zwischenspeichert, werden Download-Anfragen durch Konsumenten nicht erfolgreich beantwortet. Allerdings kann auf verschiedene Weise der Kauf eines Produktschlüssels ermöglicht werden. Dabei sollten Anbieter bei der Registrierung ihrer Softwareprodukte eine Liste mit einer festgelegten Mindestmenge gültiger Produktschlüssel im PS hinterlegen. Diese Notfall-Liste wird nur dann zu Beantwortung von Kaufanfragen genutzt, falls der betroffene Anbieter nicht erreichbar ist. Neigt sich die Anzahl verfügbarer Einträge dem Ende entgegen, wird eine neue Liste vom Anbieter angefordert. Bei anhaltender Nicht-Erreichbarkeit und fehlenden Einträgen in der Produktschlüsselliste wird das Produkt für den

Konzeption einer Erweiterung zur elektronischen Software-Distribution

entsprechenden Zeitraum vom Verkauf ausgeschlossen. Eine Freischaltung ist erst dann wieder möglich, wenn der Anbieter wieder erreichbar ist und eine neue Liste geliefert hat und somit wieder Produktschlüssel innerhalb des PS existieren.

Die zweite Möglichkeit der Auslieferung gültiger Produktschlüssel stützt sich ebenfalls auf die Bereitstellung einer Liste. Im Gegensatz zum vorherigen Szenario würde der Anbieter die Liste nicht auf seinem Server zu liegen haben, sondern sie von vornherein an das PS übermitteln. Kaufanfragen würden dann vom PS beantwortet und nicht mehr bis zum Anbieter weitergeleitet werden. Auch hier gilt, dass der Anbieter erreichbar bleiben muss. Zum einen, um mögliche Download-Anfragen zu beantworten. Zum anderen muss sichergestellt werden, dass das PS immer eine genügende Anzahl gültiger Produktschlüssel zur Verfügung gestellt sind. Um dies sicher zu stellen, sollte der Anbieter grundsätzlich auf seinem Server noch mindestens eine weitere Liste bereitstellen. Es ist immerhin möglich, dass für ein Produkt eine solche hohe Kaufbereitschaft entsteht, dass die Menge der Kaufanfragen nicht mit nur einer Liste abgedeckt werden kann. Hierbei ist der Verwaltungsaufwand auf Seiten des PS sehr hoch, jedoch können Kaufanfragen schneller beantwortet werden, da keine Kommunikation mit den Anbietern stattfinden muss.

Eine weitere Möglichkeit wäre der Upload des Produktschlüssel-Generators eines jeden Anbieters. Da jeder Anbieter seinen eigenen Mechanismus zur Verfügung stellt, wäre der Verwaltungs- und administrative Aufwand enorm. Das PS ist nicht für solche eine Aufgabe konzipiert worden. Es handelt sich nach wie vor um ein System zur Auslieferung verschiedener virtueller Waren. Die elektronische Softwareverteilung erweitert nur die bestehenden Dienstleistungen. Daher wird diese Möglichkeit nicht weiter in Betracht gezogen.

Unabhängig davon, welche Möglichkeit zur Übermittlung eines Produktschlüssels gewählt wird, muss eine Synchronisation der ausgelieferten Schlüssel auf Anbieter- und PS-Seite stattfinden. Es bleibt einem Anbieter immer freigestellt, seine Waren über andere Vertriebskanäle zum Verkauf anzubieten. Für diesen Fall kann natürlich keine 100-prozentige Unterstützung vom PS erwartet werden. Der Kommunikationsaufwand kann und sollte gering gehalten werden. Für den ersten Ansatz besteht die Möglichkeit, dass die Produktschlüssel auf Seiten des PS einen anderen Wertebereich haben, als die vom Anbieter direkt ausgelieferten Schlüssel. Somit besteht keine Gefahr, dass die verwendeten Schlüssel identisch sein könnten. Möglich wäre auch die Übermittlung einer Liste mit wenigen „Generalschlüsseln“. Diese Option sollte nur gewählt werden, wenn der Anbieter es so wünscht und die verwendeten Freischaltmechanismen dies zulassen. Anbieter sollten dann einen Produktaktivierungsmechanismus integrieren, um illegalen Installationen vorbeugen zu können.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Beim zweiten Ansatz sollte die Anzahl der Einträge pro Liste bei mindestens 100 gültigen Schlüsseleinträgen liegen. Besser wäre eine statistische Überprüfung, wie viele Download-Anfragen gestellt wurden. Auf Basis dieser Daten könnte die entsprechend Anzahl an Einträgen angefragt werden. Es ist zu Wünschen, dass das auf Anbieterseite verwendete Verwaltungssystem für ausgelieferte Produktschlüssel und -Listen korrekt funktioniert, so dass doppelte Auslieferungen ausgeschlossen werden können. Da solch ein Fall immer mal vorkommen kann, überprüft das PS die übermittelten und verwendeten Daten.

Bisher wurde davon ausgegangen, dass der Anbieter sein Produkt als Vollversion zu Testzwecken zur Verfügung stellt. Dies entspricht dem ursprünglichen Shareware-Gedanken. Jedoch ist es natürlich möglich, dass ein Anbieter nicht das ganze Programm anbieten will, sondern nur einen funktionsbegrenzten Basis-Teil. Alle weiteren Funktionalitäten sind in Module gekapselt. Erst nach erfolgreich verlaufendem Kaufvorgang werden sie an den Käufer samt Produktschlüssel ausgeliefert. Die Aufspaltung in mehrere Module lässt dem Anbieter die Möglichkeit, die Konsumenten den Funktionsumfang selbst wählen zu lassen. Je nach Wahl erfolgt die Vergütung und entsprechende Auslieferung. Ein Beispiel eines solchen Konzepts ist die in (8) vorgestellte *Game-Feature-Platform* der 4FO AG. Die Aufgabe des PS erweitert sich dahingehend, als das nicht nur der Kauf und die Auslieferung des Produktschlüssels vermerkt würden, sondern ebenfalls die mitgelieferten Module. Möchte ein Konsument die schon bestehenden Funktionsmodule ergänzen, so muss dies dem Anbieter mitgeteilt werden. Die nach Bezahlung weiteren ausgelieferten Module müssen ebenfalls vermerkt werden. Da dies einen unzumutbar hohen Verwaltungsaufwand darstellt und nicht den Anforderungen entspricht, wird diese Art des Vertriebs von vornherein ausgeschlossen.

In allen Szenarien erweitert das PS seine Rolle als Auslieferer virtueller Waren. Zusätzlich zu dem bereits bestehenden nutzerunterstützenden Informationsangebot, wird die Verwaltung gültiger Produktschlüssellisten exklusiv für Software-Anbieter als eine Dienstleistung, die als Teil eines Customer-Relationship-Management-Ansatzes (CRM) angesehen werden kann, angeboten. Dieses Angebot eröffnet Anbietern, sich an das PS zu binden und Teil der „Community“ zu werden und zu unterstützen. Mittels technisch einfacher Abläufe und komfortabler Bedienung bei der Produktregistrierung und Verkauf, sowie eine gute Sicherheit gegen Produktpiraterie, kann das PS diese Bindung vertiefen. Die daraus evtl. entstehenden Synergien können sich positiv für alle Beteiligten auswirken, inklusive der Konsumenten.

5.3.2 Auslieferung der Software als „Try-and-Die“-Version

Dieser Ansatz benutzt als Testversion eine vom Programm abgeleitete Demo-Version. Der Konsument muss im PS registriert und eingeloggt sein, um einen Download zu starten. Die

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Demo wird im öffentlichen Verzeichnis des Anbieters bereitgestellt, die Vollversion als auch die gültigen Produktschlüssel innerhalb seines geschützten Verzeichnisses. Ob sie in einem Verzeichnis oder in verschiedenen Unterverzeichnissen gehalten werden ist dem Anbieter zu überlassen.

Anders als bei der Auslieferung einer Shareware-Version, ist die Demo nach Ablauf ihrer Beschränkungen nicht mehr ausführbar. Entschließt ein Konsument das Produkt zu kaufen, so muss die Vollversion und ein dazugehöriger Schlüssel ausgeliefert werden. Der Anbieter kann sich entscheiden, ob die Vollversion einheitlich für alle Konsumenten angeboten wird oder ob jeder Käufer ein personalisiertes Programm erhält.

Die Variante einer einheitlichen Version für alle Konsumenten entspricht prinzipiell den vorhandenen Abläufen beim Erwerb registrierter Musikdateien. Der Käufer lädt sich die Vollversion herunter und erhält zusätzlich einen Produktschlüssel. Der Anbieter weiß nur, dass ein Produkt samt Schlüssel verkauft und ausgeliefert wurde. Für ihn ist der Käufer anonym.

Technisch anspruchsvoller wäre eine personalisierte Version. Dazu werden Informationen benötigt, die die Anonymität der Käufer aufweichen. Welche Informationen im speziellen vom Anbieter verlangt werden, hängt von seinen vorhandenen Mechanismen zur Generierung einer personalisierten Programmversion ab. Jedoch sind von Seiten des PS Grenzen gesetzt. Der Datenschutz bleibt soweit vorhanden, als dass keine persönlichen Daten wie Name oder Anschrift des Käufers preisgegeben werden dürfen und sollen. Weniger kritisch wäre die Übermittlung der internen Benutzeridentifikationsnummer. Ein Anbieter kann damit den Wertebereich möglicher Produktschlüssel für Vollversionen pro Benutzer einschränken. Somit wäre ebenfalls der Code zum Freischalten personalisiert. Nur dieser Käufer kann nochmals einen Produktschlüssel anfragen. Falls dies geschieht, wird die vom PS gelieferte Identifikationsnummer zu Authentifizierung herangezogen. Ist sie identisch, kann ein neuer Schlüssel generiert und über das PS ausgeliefert werden.

Die vom Anbieter verarbeiteten Daten werden derart in die Vollversion integriert, dass das ausgelieferte Programm im für den Konsumenten schlimmsten Fall nur auf einem Computer ausführbar ist. Hier kann eine Analogie zu *Digital-Rights-Management* (DRM) Systemen im Musikbereich gezogen werden. So können bei diesem Ansatz ebenfalls Probleme bei der Ausführung legal erworbener Programme entstehen, z.B. wenn der Konsument sich neue Hardware zulegt oder ein komplett neues Computersystem. Solch gravierende Änderungen können nachträglich nicht mehr vom Programm berücksichtigt werden. Somit müsste der Konsument sich eine erneute Version kaufen bzw. den Anbieter kontaktieren und den Sachverhalt erklären und eine neue Version verlangen.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Dies ist keine Art und Weise mit den Konsumenten umzugehen, noch kann man ihnen solch einen Aufwand zumuten. Daher sollte diese Art der Verteilung nicht weiter verfolgt werden oder nur unter dem Gesichtspunkt, dass die herangezogenen Daten nicht die Ablauffähigkeit des Programms auf einen Computer beschränken. Man kann es einem Konsumenten eher noch zumuten, auf seinem Computersystem einen bestimmten Accountnamen immer wieder verwenden zu müssen. Das Programm kann nur unter diesem Namen ausgeführt werden.

Das Anbieten einer Demo-Version mit anschließendem Kauf einer Vollversion inklusive gültigem Produktschlüssel ist für den Anbieter ein relativ sicheres Konzept. Doch selbst eine Demoversion kann von professionellen Hackern in eine Vollversion umgewandelt werden, falls sie nicht funktionslimitiert ist. Der Aufwand für Käufer, die zwei verschiedene Programmversionen (Demo und Original) kopieren zu müssen, ist zwar aus technischer Sicht mit heutigen Breitbandanschlüssen relativ gering. Doch besitzt nicht jeder Konsument einen entsprechenden Anschluss. Des Weiteren ist die Übertragung eines einzelnen Produktschlüssels schneller als ein komplettes Programm.

Die Möglichkeiten einer personalisierten Versionsauslieferung sind vielfältig und technisch realisierbar, aber entsprechen durch evtl. auftretende Einschränkung auf Nutzerseite nicht der Motivation und dem Sinn des PS. Hoch entwickelte ESD-Systeme wie der Firma PACE (52) unterstützen diese Möglichkeit der Bindung von Programmen und Computersystem.

5.4 Erweiterungen des PotatoSystems

Die Überlegungen im vorherigen Kapitel sollen als Grundlage dienen, um die möglichen Veränderungen am PS genauer zu beschreiben. Anforderungen werden ebenfalls an die Anbieter gestellt, um ein funktionierendes ESD zu gewährleisten. Auch diese werden genauer beschrieben.

Die Modifizierungen und Erweiterungen auf Seiten des PS können je nach Ansatz sehr unterschiedlich ausfallen. Im Folgenden wird aus den genannten Gründen in 5.3.2 nur auf das Konzept aus 5.3.1 eingegangen.

5.4.1 Auslieferung

Zunächst soll der Auslieferungsteil des PS betrachtet werden. Die Anfrage einer Datei auf einem Anbieter-Server kann beim gewählten Shareware-Ansatz die bisherigen Strukturen und Abläufe verwenden. Einziger Unterschied ist, dass Softwareprogramme nicht temporär im Proxy-Cache des Jackets gespeichert werden sollen. Das Dispatcher-System braucht somit nicht mehr den Cache zu durchsuchen, sondern kann Anfragen gleich an den Proxy weiterleiten. Der wird seinerseits in einen transienten Modus geschaltet. Somit bilden Dispatcher und

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Proxy zusammen eine Art Tunnel zwischen den Systemen der Anbieter und Käufer, durch den die angefragten Daten übermittelt werden.

Die Hauptüberlegung ist nun der Zugriff auf den Inhalt des geschützten Verzeichnisses des Anbieters. Dort liegen die Listen mit gültigen Produktschlüsseln. Da Anbieter mehrere Produkte zum Verkauf anbieten können, müssen die Listen eindeutig einem Programm zugeordnet werden können, was im folgenden Kapitel 5.5 beschrieben wird. Es empfiehlt sich eine modulare, gekapselte Lösung, die unabhängig vom bestehenden System agieren kann. Die bereitgestellte Funktionalität kann zum Zeitpunkt einer Kaufanfrage aufgerufen werden. Die Erweiterung ist somit minimal-invasiv. Die folgende Abbildung 7 zeigt ein Architekturbild des PS inklusive der ESD-Erweiterung.

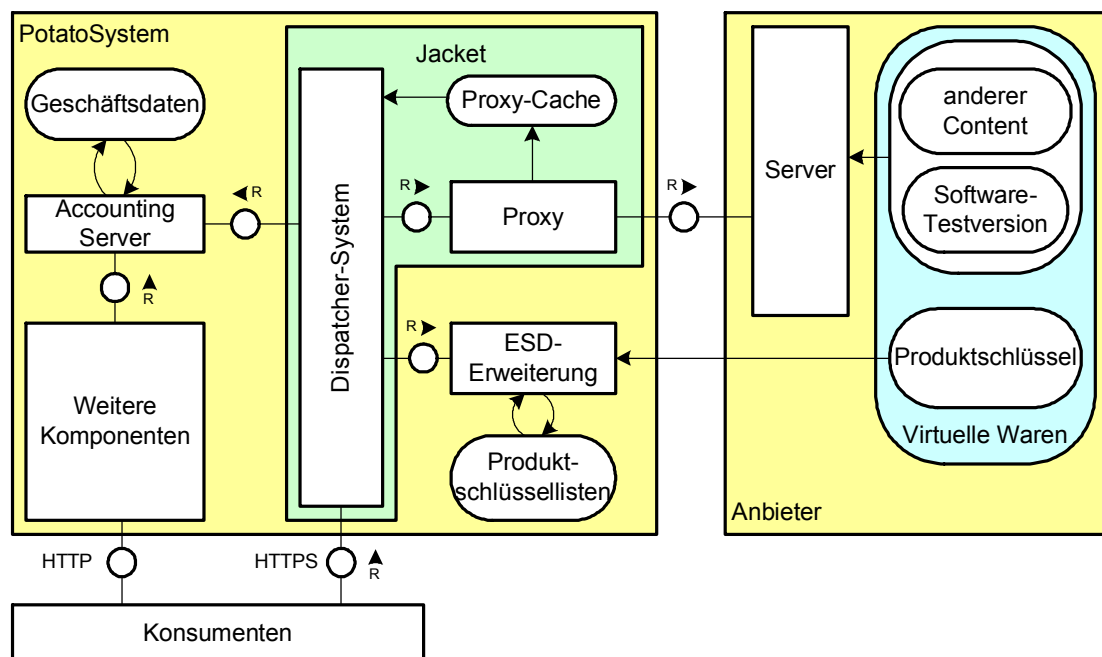


Abbildung 7 ESD-Erweiterung im PotatoSystem

Es ist zu sehen, dass die ESD-Erweiterung unabhängig vom bestehenden PS agiert. Nur sie kann Zugriff auf die Produktschlüssel erlangen. Damit die Daten aus dem Anbieterverzeichnis tatsächlich ihren Weg zum PS finden, wird den Anbietern ein Skript zur Verfügung gestellt, dass sie zusammen mit den Produktschlüsseln in das geschützte Verzeichnis kopieren. Das Skript soll den Zugriff und die Verwaltungsmöglichkeiten sowohl auf Seiten des PS als auch der Anbieter vereinfachen. Auf dieses Skript wird im folgenden Kapitel 5.5 weiter eingegangen. Auf alle weiteren Inhalte wird nach bestehenden Mechanismen zugegriffen.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

5.4.2 Verwaltung

Der Verwaltungsteil der ESD-Erweiterung findet an verschiedenen Orten innerhalb des PS statt. Hauptsächlich geht es um die Verwaltung von Produktschlüsseln. Damit zusammenhängen Maßnahmen bei Nicht-Verfügbarkeit eines Anbieters. Der Umgang mit den Produktschlüssellisten ist ein zentraler Bestandteil auf Seiten des PS. Da die ESD-Erweiterung unabhängig vom Rest des PS arbeiten soll, können die Listen nicht im Proxy-Cache gehalten werden. Das wäre auch nicht zweckmäßig, da der Cache für Dateien gedacht ist, die als Ganzes angefragt werden. Das ist bei Produktschlüssellisten nicht der Fall. Des Weiteren kann der Cache geleert werden und somit wären sämtliche Produktschlüssel verloren. Daher ist eine eigenständige Verwaltung nötig, die eine schnelle Auslieferung der Schlüssel begünstigt, dabei die Integrität und Konsistenz der Liste nicht verletzt. Die Lösung ist abhängig vom gewählten Ansatz der Schlüsselauslieferung.

5.4.2.1 Verwendung einer Notfallliste

In dieser Variante muss die Liste nur benutzt werden, wenn der Anbieter nicht erreichbar ist oder keine Produktschlüssel mehr liefert. Der Anbieter hat nach der Produktregistrierung eine Notfallliste mit mindestens 20 gültigen Produktschlüsseln zu übergeben. Im Notfall wird die Datei exklusiv für nur eine Anfrage geöffnet, ein Eintrag gelesen, kopiert und daraufhin gelöscht, um ihn nicht ein weiteres Mal zu verwenden. Somit steht am Anfang der Datei ein gültiger, unbenutzter Eintrag. Der Eintrag wird nur dann entnommen, wenn die Bezahlung durch den Konsumenten erfolgreich war. Er wird dann in einer nur für den Zweck einer ESD erstellten Datenbank-Tabelle gespeichert, auf die weiter unten eingegangen wird. Das bedeutet, dass der Produktschlüssel zwar aus der Liste entfernt ist, aber im PS noch existiert. Dies ist wichtig, da der von einem Konsumenten erworbene Schlüssel immer auf dessen Transaktionsseite angezeigt werden soll, als auch für die Eindeutigkeitsüberprüfung.

Jede vorhandene Liste besitzt nur eine begrenzte Anzahl von Einträgen. Sollte bei einer Anfrage festgestellt werden, dass der Anbieter nicht verfügbar ist und die Menge vorhandener Schlüssel unter einen willkürlich ausgewählten Wert fällt, sind zwei Aufgaben durchzuführen. Zum einen muss der Anbieter davon in Kenntnis gesetzt und die Bereitstellung einer neuen Liste gefordert werden, bspw. mittels einer E-Mail. Zum anderen muss in regelmäßigen Intervallen seine Verfügbarkeit geprüft werden. Die einfachste Möglichkeit ist das Senden eines Ping-Signals. Die Häufigkeit ist abhängig von der Anzahl an noch vorhandenen Einträgen in der Liste. Je weniger es sind, desto häufiger muss das Signal gesendet werden. Ein Vorschlag ist in der folgenden Tabelle 1 aufgelistet.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Einträge	Intervall in Minuten
≥ 20	-
11-19	60
5-10	20
< 5	5

Tabelle 1 Intervall des Ping-Signals

Ist der Anbieter wieder erreichbar, bevor die Liste geleert ist und sind weniger als 20 Einträge vorhanden, kopiert das PS die neue Reserveliste aus dem geschützten Verzeichnis und hängt die Einträge an das Ende der existierenden Datei.

5.4.2.2 Verwendung von Auslieferungslisten

Der Ablauf für die Variante, in der das PS für die Auslieferung vollständig zuständig ist, ändert sich teilweise. Der Anbieter wird weiterhin bei jeder Downloadanfrage des Produkts kontaktiert. Käuferanfragen für gültige Produktschlüssel werden vom PS beantwortet. Auch hier muss der Anbieter bei der Registrierung seiner Ware eine Liste übergeben, die mit mindestens 100 gültigen Produktschlüsseln ausgestattet sein sollte. Je mehr Einträge die Liste besitzt, desto länger hält eine Liste vor. Das bedeutet, dass das PS weniger Anfragen nach neuen Listen starten muss. Somit kann die Kommunikation zwischen Anbieter und PS minimal gehalten werden. Kaufanfragen werden auch bei Nicht-Verfügbarkeit des Anbieters beantwortet, solange noch Einträge in der Liste stehen. Auch hier müssen die Integrität geprüft, die interne Darstellung und persistente Speicherung wie oben beschrieben durchgeführt werden.

Zur schnelleren Beantwortung konkurrierender Anfragen kann in Erwägung gezogen werden, die vorhandene Liste in mehrere kleine Teile zu zerlegen. Beispielsweise kann eine Liste mit 100 Einträgen in fünf Listen á 20 Einträge geteilt werden. Davon können vier für die Beantwortung von Schlüsselanfragen sofort genutzt werden. Eine Liste wird zurückgelegt und erst dann verwendet, wenn alle anderen aufgebraucht worden sind. Wird der erste Eintrag angefordert wird gleichzeitig eine neue Liste beim Anbieter angefragt. Diese sollte schon im geschützten Verzeichnis existieren und sofort auslieferbar sein. So kann gewährleistet werden, dass selbst bei einer hohen Anzahl konkurrierender Schlüsselanfragen durch Konsumenten eine zuverlässige Auslieferung verfügbar ist.

5.4.3 Speicherformat der Produktschlüssellisten

Hier stellt sich die Frage, wie die Listen intern im PS gespeichert werden. Die einfachste Lösung wäre eine simple ASCII-codierte Textdatei.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Der Einfachheit halber sollte der Anbieter ebenfalls solch eine Textdatei für jedes seiner Produkte bereitstellen. Jeder Produktschlüssel entspricht einem einzeiligen Eintrag.

Das PS bietet auch die Möglichkeit CSV-Dateien zu nutzen. Dabei handelt es sich um einfache Textdateien, deren Einträge mittels eines speziellen Trennzeichens unterschieden werden. Als Trennzeichen kommen das Komma, das Semikolon, der Doppelpunkt und der Tabulator in Frage²⁰. In CSV-Dateien können Listen von Listen innerhalb einer einzelnen Datei existieren. Jede Zeile repräsentiert eine Liste. Einen Standard gibt es nicht. Bisher wird dieser Mechanismus vom PS nur verwendet, um mehrere virtueller Waren mittels Import auf einmal zu registrieren und durch Export Anbietern eine bequeme Offline-Verwaltung ihrer Verkäufe zu ermöglichen. Es spricht nichts dagegen, eine CSV-Datei mit gültigen Produktschlüsseln vom Anbieter zu erhalten. Die Einträge müssen nur entsprechend konvertiert werden, d.h. das jedem Wert nachfolgende Trennzeichen wird durch einen Zeilenumbruch ersetzt. Andererseits kann das CSV-Format verwendet werden, um die Produktschlüssellisten eines einzelnen Anbieters in einer Datei unterzubringen. Dies würde nicht nur Speicherplatz sparen, sondern ebenfalls die Verwaltung vieler Dateien auf eine einzige oder einige wenige beschränken. Jedes registrierte Produkt würde dann einen einzeiligen Listeneintrag mit allen dem PS übergebenen Produktschlüsseln erhalten. Der entscheidende Nachteil ist hierbei die Beantwortung konkurrierender Kaufanfragen. Die Datei wird für jede Anfrage exklusiv geöffnet, um die Dateikonsistenz zu wahren und fehlerhafte Auslieferungen, z.B. zweimal den gleichen Produktschlüssel, zu vermeiden. Dies betrifft alle Anfrage die an den betroffenen Anbieter gerichtet werden, nicht nur die für ein einzelnes Produkt. Eine schlechte Performanz wäre die Folge. Der Kompromiss zwischen Datenkonsistenz, Speicherplatzverbrauch und Performanz der Beantwortung von Anfragen muss genau überlegt werden. So ist dieser Ansatz eher für die Verwendung einer Notfallliste zu gebrauchen als für die ständige Auslieferung der Produktschlüssel durch das PS.

Unzweckmäßig ist ebenfalls die Verwaltung der Produktdaten mehrerer Anbieter innerhalb einer CSV-Datei. Auch wenn es unwahrscheinlich ist, dass mehrere Anbieter gleichzeitig nicht verfügbar sind, sollte dieser Fall trotzdem bedacht und mit einkalkuliert werden. Hier treten die im vorherigen Absatz erläuterten Probleme in verschärfter Form wieder auf. Daher ist die Verwaltung der Produktdatensätze mehrerer Anbieter innerhalb einer Datei nicht zweckmäßig.

Eine andere Repräsentationsmöglichkeit wäre die Erstellung einer Datei, deren Inhalt im XML-Format gespeichert werden würde.

²⁰ CSV-Dateien können mit einfachen Texteditoren gelesen werden oder durch spezielle Tabellenkalkulationsprogramme wie Microsofts Excel, welches als Trennungszeichen das Semikolon verwendet.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Sämtliche Produktschlüssel eines Anbieters und dessen Produkte könnten innerhalb einer einzigen Datei verwaltet werden. Diese Lösung hat allerdings drei entscheidende Nachteile. Zum einen ist der Speicherplatzverbrauch einer solchen XML-Datei enorm und unzweckmäßig hoch. Das betrifft sowohl den Speicherplatz auf einem Datenträger als auch im Arbeitsspeicher eines Computersystems. Zum zweiten sind die Zugriffsgeschwindigkeiten relativ lang, vor allem wenn es sich um Schreibzugriffe handelt. Diese sind allerdings nötig, um die verwendeten Einträge zu entfernen. Drittens ist wieder das Problem konkurrierender Kaufanfragen gegeben, welches in den beiden vorherigen Absätzen besprochen wurde. Somit ist die Verwaltung mittels XML-Dateien als nicht zweckmäßig anzusehen, egal ob jeder Anbieter einzeln oder alle in einer Datei verwaltet würden.

5.4.4 Identifizierung der Produktschlüssellisten

Die bei der Registrierung an das PS übertragene Liste wird intern von der ESD-Verwaltung umbenannt, um eine eindeutige Zuordnung zwischen Liste und Produkt zu gewährleisten. Dazu wird die eindeutige Datei-Identifikationsnummer des Produkts (FileID) verwendet. Im Fall einer Notfallliste erhält jede Liste zusätzlich noch das Suffix „*internalEmergencyList*“, um ihre Verwendung eindeutig zuweisen zu können. Der Dateiname einer solchen Liste würde dann wie folgt lauten: „*FileID_internalEmergencyList*“. Im anderen Auslieferungsfall wird das Suffix „*internalPKList*“ verwendet. Sollte eine Aufspaltung dieser Liste in mehrere Teillisten durchgeführt werden, so wird noch eine Teilnummer mit „00“ beginnend an den vorhandenen Namen gehängt. Ein Beispiel wäre „*FileID_internalPKList02*“.

5.4.5 Persistente Speicherung verwendeter Produktschlüssel

Die zweite wichtige Verwaltungsaufgabe des PS besteht in der Zuordnung von Produktschlüssel und durchgeführter Transaktion. Bisher wurde nach erfolgreicher Bezahlung einer virtuellen Ware die generierte TAN in den Namen der ausgelieferten Datei eingefügt. In 5.2 wurde kurz erläutert, dass dieses Verfahren bei Softwaredistribution vermieden werden sollte. Trotzdem muss es eine Zuordnung der Transaktion zum Verkauf eines Produkts geben. Da eine Datei nur einmal herunter geladen zu werden braucht, Konsumenten aber mehrere Produktschlüssel erwerben können, empfiehlt es sich, eine Abbildung zwischen diesen beiden Werten herzustellen. Das PS muss zudem einen zur Auslieferung bestätigten Produktschlüssel speichern. Zum einen, um die Anzahl angefragter Schlüssel berechnen zu können und sie auch dem Käufer auf seiner Transaktionsseite anzeigen zu können. Zum anderen zur Eindeutigkeitsüberprüfung, um doppelte Schlüsselauslieferungen zu vermeiden.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Daher ist es zweckmäßig, eine neue Datenbanktabelle anzulegen, die vom Accounting-Server (AS) verwaltet wird. Die Tabelle besteht nur aus zwei Spalten

PKOwnership_as	
	PK
FK	TAN

Tabelle 2 ESD-spezifische Datenbanktabelle

Der Tabellename PKOwnership_as soll ihre Aufgabe repräsentieren. PK steht dabei für ProductKey (Produktschlüssel), TAN für Transaktionsnummer, die gleichzeitig ein Fremdschlüssel (FK) aus einer weiteren Tabelle ist. Das Suffix „_as“ ist ein Hinweis, dass die Tabelle vom AS verwaltet wird. Beide Spalten bilden zusammen den Tabellenschlüssel. Eine zusätzliche Überlegung wäre, ob die Identifikationsnummer des Anbieters als weitere Spalte in die Tabelle eingefügt werden sollte. So könnten Anfragen von Anbietern, die ihre Verkaufsstatistik erstellt haben möchten, theoretisch schneller beantwortet werden. Dies müsste in Performance- und Lasttests geprüft werden.

5.4.6 Produktregistrierung und -Updates

Registriert ein Anbieter sein Produkt, ermittelt das PS einen Hash-Wert, der persistent in einer Datenbank gespeichert wird. Im Gegensatz zu Musikstücken unterliegen Softwareprogramme in den meisten Fällen einer fortwährenden Entwicklung. Modifizierungen an Softwareprodukten sind keine Seltenheit. Dadurch entstehen allerdings andere Hash-Werte als die bei der ursprünglichen Registrierung. Damit dieser Entwicklung Rechnung getragen werden kann, muss die Registrierung von Software komfortabler und flexibler gestaltet werden als die von Musikstücken. Anbieter müssen die Möglichkeit erhalten, modifizierte Programme erneut zu registrieren, ohne dabei sämtliche Daten wiederholt eingeben zu müssen. Eine Möglichkeit wäre, auf der persönlichen Transaktions-Web-Seite des Anbieters die Option auf ein Update eines bereits registrierten Produkts bereit zu stellen. Die bereits gespeicherten Daten werden übernommen. Intern werden ein neuer Hash, eine neue FileID als auch eine neue TAN erzeugt. Dazu sollte eine Option angeboten werden, die bereits verwendete/n Produktschlüssel-liste/n zu nutzen. Falls nicht, wird mit der Registrierung fortgefahren, als würde es sich um ein neues Produkt handeln.

Jeder Konsument erhält vom PS nur komplette Softwareprogramme zum Testen und Kauf angeboten. Das bedeutet, dass bereits erworbene Produkte nur durch Kommunikation mit dem Anbieter ein Update erhalten können. Schließlich ist das PS nicht dafür zuständig, einzelne Programmfragmente zum Kauf anzubieten.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

5.5 Anforderungen und Erweiterungen auf Anbieterseite

Die Anbieter haben einige Voraussetzungen zu erfüllen, um einen Verkauf ihrer Produkte zu ermöglichen. Zuerst muss die Verzeichnisstruktur zur Auslieferung virtueller Waren nach den Regeln des PS auf ihrem Server hergestellt werden. Das bedeutet, dass es ein öffentliches und ein geschütztes Verzeichnis erstellt werden müssen. Dabei ist es nicht erforderlich, dass die Verzeichnisse auf demselben Server zu finden sind. Wichtig ist nur, dass die Verweise bei der Registrierung korrekt eingetragen werden. Wie schon weiter oben mehrmals erwähnt, wird die Testversion in das öffentliche Verzeichnis gelegt, die zugehörigen Produktschlüssel in das geschützte.

Der Anbieter hat die Aufgabe, dass zu verkaufende Programm als Sharewareversion anzubieten. Welcher Mechanismus verwendet wird steht ihm frei. So kann dieser bereits ins das Programm hineincodiert sein oder er verwendet ein softwarebasiertes Werkzeug, das die ausführbare Datei in eine „Prüf-vor-Kauf“-Version ändert. Die meisten dieser Werkzeuge ummanteln das Programm, weshalb sie als *Wrapper* bezeichnet werden oder verlangen die Verwendung spezieller Bibliotheken.

Das PS will seine Anbieter soweit unterstützen, dass mindestens ein solches Werkzeug zur Verfügung gestellt wird bzw. es einen Verweis auf die Anbieterseite dieses Programms gibt. Im Anhang A wird solch ein Tool vorgestellt und kurz die Generierung einer Shareware beschrieben.

5.5.1 Bereitstellung der Produktschlüssellisten

Unabhängig des gewählten Mechanismus ist die Bereitstellung einer Liste mit gültigen Produktschlüsseln notwendig. Wie schon im vorherigen Kapitel 5.4.4 erwähnt, müssen mindestens zwei Listen zusammengestellt werden. Eine wird bei der Registrierung des dazugehörigen Produkts ans PS übermittelt, die andere verbleibt bis zur Anfrage im geschützten Verzeichnis.

5.5.1.1 Identifizierung

ES stellt sich die Frage der Identifikation und eindeutigen Abbildung einer Liste zu ihrem korrespondierenden Programm. Für die zum Upload bereitgestellte Liste wird verlangt, dass sich ihr Name aus Produktname oder, falls vorhanden, Produktnummer und dem Suffix „*_PSUpload*“ zusammensetzt. Sollten von Beginn an mehrere Listen existieren, so können zusätzliche Nummerierungen benutzt werden, z.B. „*_PSUpload00*“, „*_PSUpload01*“, usw. Für die im Verzeichnis verbleibenden Listen gibt es nun verschiedene Identifizierungsmög-

Konzeption einer Erweiterung zur elektronischen Software-Distribution

lichkeiten. Die intuitivste Lösung wäre, der Liste den Namen des Produkts zuzuweisen. Dieser doch eher schwache Mechanismus kann durch die Verwendung von evtl. vorhandenen Produktnummern verstärkt werden. Doch schließt das nicht aus, dass zwei Produkte verschiedener Anbieter die gleiche Produktnummer besitzen. Daher bietet es sich an, den Produktnamen gefolgt von der Produktnummer zu verwenden, falls letztere existiert. Da dies nicht immer der Fall sein muss, sollte ein vom PS generierter Namenszusatz integriert werden. Verwendung kann die bei der Registrierung generierte TAN oder Datei-Identifikationsnummer (FileID) finden. Sie sind innerhalb des PS eindeutig und wiederholen sich nicht. Der bisherige Registrierungsablauf sieht vor, dass einem Anbieter die TAN integriert im Verkaufslink mitgeteilt wird. Für die TAN spricht, dass sie die Berechnungsgrundlage aller Provisionen und Verkaufserlöse ist. Allerdings stellt dies bei der Identifizierung der Liste keinen hinreichenden Grund dar, weshalb nicht die im PS intern generierte FileID herangezogen werden sollte. Sie ist ebenso einfach mitzuteilen wie die TAN, braucht aber nicht in einen Verkaufslink integriert dargestellt zu werden, sondern kann einem Anbieter auf seiner Transaktions-Webseite als einfacher Eintrag aufgelistet werden. Unabhängig davon, sollte die PS-generierte Zeichenkette als Suffix in den Listennamen integriert werden oder gar als einziger Identifizierungsmechanismus. Eine Produktschlüsselliste würde also folgendermaßen identifiziert werden: „*[productName]TAN*“ bzw. „*[productName]FileID*“. Eine derartige Verwendung inkludiert, dass solch eine Liste nur einmal im Verzeichnis existieren darf. Diese Vorgabe bedeutet zwar einen kleinen Verlust der Entscheidungsfreiheit

5.5.1.2 Einheitlicher Zugriff

Die Übermittlung eines einzelnen Produktschlüssels als auch einer Produktschlüsselliste an das PS soll für alle Anbieter dem gleichen Ablauf folgen. Daher erhält jeder ein Skript, welches speziell für diese Aufgabe konstruiert wurde. Neben dem einheitlichen Ablauf ist dies eine Vereinfachung der Handhabung für die Anbieter. Sie müssen das Skript zusammen mit den Listen in das geschützte Verzeichnis zu kopieren. Dabei ist es unabhängig, ob alle Listen jedes registrierten Produkts zusammen in einem geschützten Verzeichnis liegen oder für jede Liste ein eigenes geschütztes Verzeichnis angelegt wird. Letzteres verlangt, dass in jedem Verzeichnis das Skript hineinkopiert wird.

Fragt das PS eine neue Notfallliste an, so ist es die Aufgabe des Skripts, die entsprechende Datei zu übertragen. Dazu muss das PS als Parameter den eindeutigen Listennamen übermitteln. Den Ansatz verfolgend, dass die Anbieter für jede Kaufanfrage eines Produktschlüssels kontaktiert werden, hat das Skript auf dieser Seite die gleiche Verwaltungsaufgabe wie das

Konzeption einer Erweiterung zur elektronischen Software-Distribution

PS. Die Datei, in der die Produktschlüssel gespeichert sind muss exklusiv geöffnet, der erste Eintrag gelesen, kopiert, an das PS übertragen und aus der Liste gelöscht werden. Darüber hinaus muss es eine Überprüfung der Anzahl verfügbarer Einträge geben. Sollten nur noch wenige vorhanden sein, muss dies dem Anbieter mitgeteilt werden. Aus diesen Gründen muss die verwendete Skriptsprache über einen entsprechenden Funktionsumfang verfügen. Zusätzlich ist zu bedenken, dass nicht jeder Anbieter die administrative Kontrolle über seinen File-Server besitzt. Ein Hobby-Entwickler kann seine Programme auf einem Server eines speziellen Web-Hosting-Anbieters (z.B. Strato, 1&1 Internet AG, etc.) kopieren und von dort zum Verkauf anbieten. Da dieser Fall aller Wahrscheinlichkeit nach häufig vorkommt, muss die gewählte Skriptsprache ebenfalls weit verbreitet sein und von den großen Host-Anbietern unterstützt werden. Gesetzt dem Fall, dass die benötigte Unterstützung nicht gegeben werden kann, muss ein direkter Zugriff auf die Daten der Produktschlüssellisten möglich sein. Allerdings entfällt die unterstützende Verwaltung durch das bereitgestellte Skript. Hier ergibt sich nun wiederum das Identifizierungsproblem, denn ein Anbieter kann zum Zeitpunkt der Registrierung noch nicht die TAN oder FileID seines Produkts kennen. Daher muss es ihm gestattet werden, den im Pfad enthaltenen Dateinamen der Produktschlüsselliste zu einem späteren Zeitpunkt zu modifizieren.

5.5.2 Produkt-Updates

Wie schon im Kapitel 5.4.6 beschrieben, kann ein Anbieter bei Updates schon vorhandene Schlüssellisten verwenden. Dies ist eine sinnvolle Option, um den Verwaltungsaufwand auf Seiten des Anbieters als auch des PS gering zu halten. Auf jeden Fall müssen alle registrierten Versionen eines Produkts weiterhin verfügbar sein, da Konsumenten für eine bestimmte Version bezahlt haben. Es bleibt Anbietern vorbehalten, jedem Konsumenten die neuste Version zum Download bereit zu stellen, aber unter Umständen wollen einige Käufer diese Option gar nicht wahrnehmen, weil sie mit ihrer erworbenen Version hervorragende Ergebnisse erzielen oder ihr System auf diese Version konfiguriert ist und stabil läuft. Allerdings muss auch die Möglichkeit gegeben sein, alte Produktversionen aus dem Verkauf zu nehmen und Käufern eine höhere anzubieten.

5.6 Erstellen eines Sharewareprogramms als Dienstleistung des PotatoSystems

Das PS seine Rolle als reines Vermittlungs- und Auslieferungssystem virtueller Waren durch ein Angebot verschiedenster Dienste sowohl für Konsumenten als auch für Anbieter.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Eine weitere Dienstleistung wäre im Bereich ESD die Online-Erstellung einer Sharewareversion aus einer ausführbaren Datei und der von diesem Programm verwendeten Quellen. Diese würden einmalig in das PS per Upload kopiert und nach Fertigstellung wieder zum Anbieter zurück transferiert werden.

Es existiert eine geringe Menge an Werkzeugen, die aus einem vorhandenen Programm eine Sharewareversion erschaffen. Die Ansätze sind unterschiedlich gewählt. Einige ummanteln das Programm und fügen evtl. eine Installationsroutine hinzu. Andere setzen die Verwendung bestimmter Bibliotheken bei der Entwicklung des Programms voraus. Letztere sind für die Verwendung des betrachteten Ansatzes ungeeignet. Zum einen, weil Entwickler erst die Bibliotheken vom PS erhalten müssen. Obwohl es sich hierbei technisch um einen einfachen Vorgang handelt, kann dies zu juristischen Problemen führen. Die Bibliotheken sind für den Lizenznehmer und dessen Entwicklungsarbeit registriert und verkauft worden, in diesem Fall eine Person der 4FO AG die mit und am PS arbeitet. Die Auslieferung an Dritte darf nur mit der Einwilligung des Werkzeuganbieters geschehen. Dies würde wahrscheinlich zu einer Umsatzbeteiligung bei Verkäufen von Produkten, die diese Bibliotheken verwenden oder einem festgelegten Preis pro Auslieferung führen. Werden diese Mehrausgaben an die Anbieter weitergereicht, ist damit zu rechnen, dass sich dies im Preis der Waren niederschlagen wird. Es sei denn, die Anbieter nehmen diesen finanziellen Verlust hin. Dies würde das PS allerdings unattraktiver für den Verkauf machen. Die letzte Möglichkeit wäre, dass das PS die Kosten übernimmt und dem Anbieter nicht die üblichen fünf Prozent eines Warenverkaufs berechnet, sondern acht bis zehn Prozent. Dies entspricht allerdings der schon erwähnten unattraktiven Preiserhöhung und scheidet somit aus weiteren Überlegungen aus.

Zum anderen bedeutet die Verwendung spezieller Bibliotheken eventuell die Festlegung auf bestimmte Plattformen und Programmiersprachen. Somit wäre die Dienstleistung beschränkt und würde die Nutzer des PS, welche als Anbieter auftreten möchten, in ihrer Entscheidungsfreiheit einschränken. Auf die Frage, ob es überhaupt zweckmäßig ist, eine Vielzahl verschiedenster Technologien zu unterstützen, wird nochmals später eingegangen.

Es bleiben also nur die so genannten *Wrapper* übrig, d.h. die Werkzeuge, die das bereits erstellte Programm mit einem Schutzmechanismus versehen. Auch hier ist auf Technologie-Neutralität zu achten. Ein Beispiel ist die Verwendung des Betriebssystems der Nutzer. Diese wollen wahrscheinlich nicht nur auf ein einziges wie Microsofts Windows festgelegt sein, sondern ebenfalls unix-basierten Systeme verwenden und/oder für diese Programme entwickeln. Doch gerade *Open Source* System wie Linux werden von der Anhängerschaft mit frei erhältlichen Programmen überhäuft, so dass sich die Frage aufdrängt, ob überhaupt für diese

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Plattformen Programme zum Verkauf angeboten würden. Um trotzdem Technologie- und Plattform-Unabhängigkeit zu erhalten, können verschiedener Werkzeuge auf Seiten des PS Verwendung finden. Allerdings ist dies auch eine Kostenfrage, die sowohl ein Für als auch ein Wider dieses Ansatzes ist.

Solche Werkzeuge, die den genannten Anforderungen genügen sollen, sind in der Regel teuer, da sie mehr bieten als nur den Mechanismus, aus einer ausführbaren Datei eine Sharewareversion zu erstellen. Als Beispiel kann *FLEXnet InstallShield* als Premiumversion v12 der Firma „ComponentSource“ dienen (57). Obwohl es sich hierbei um ein auf Microsoft Windows abzielendes Produkt handelt, ist es ein hoch entwickeltes Tool, das viele zusätzliche Funktionalitäten bietet. Eine Einzelplatzlizenz kostet bei Neuerwerb 2137,- Euro. Entwickler oder kleinere Firmen können sich solche Tools evtl. nicht leisten oder es würde sich für sie finanziell nicht rechnen. Sie gehen dann einen Kompromiss zwischen Sicherheit der ausgelieferten Software vor Hackerattacken und dem vermuteten finanziellen Verlust ein. Des Weiteren müssen auch die Mitarbeiter geschult und eingearbeitet werden.

Der Ansatz sieht vor, dass für einen Entwickler des PS eine Einzelplatzlizenz erworben wird. Das verwendete Werkzeug muss dann die Möglichkeit bieten, die Sharewareherstellung bzw. Umwandlung automatisch durchführen zu lassen. Der Ablauf würde folgendermaßen aussehen:

Konzeption einer Erweiterung zur elektronischen Software-Distribution

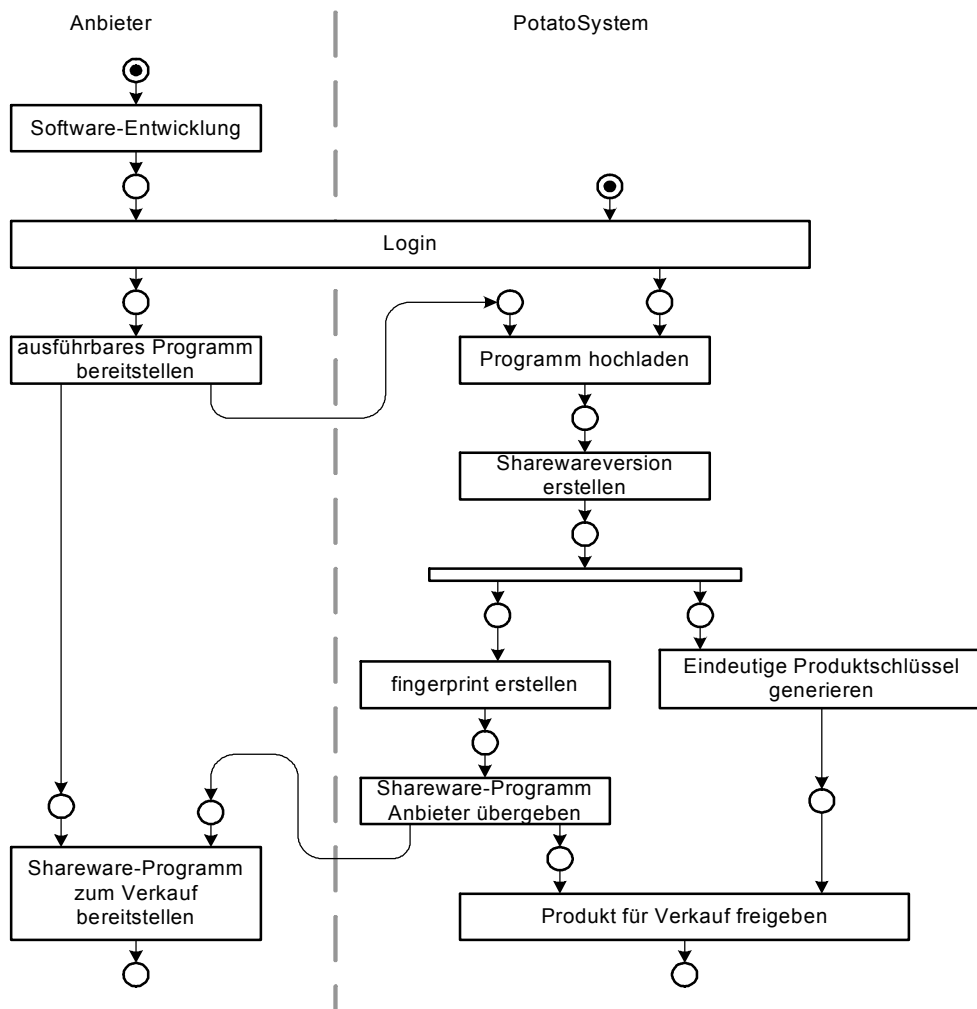


Abbildung 8 Shareware-Umwandlung als PS-Dienstleistung

Ein Anbieter möchte sein Programm im PS registrieren und entscheidet sich, den Dienst in Anspruch zu nehmen. Er gibt den Pfad an, wo das ausführbare Programm liegt. Sollten weitere Dateien zur Ausführung des Programms nötig sein, so müssen sie ebenfalls innerhalb dieses Pfades zu finden sein. Alle Dateien werden hochgeladen und in eine Sharewareversion umgewandelt. Anschließend kopiert das PS die generierte Programmversion in das angegebene öffentliche Verzeichnis auf dem Anbieterserver.

Dieser Ansatz verändert nicht nur den Inhalt eines digitalen Produkts, sondern überträgt dem PS auch die Verantwortung der Schlüsselgenerierung und deren Vergabe. Die Anbieter haben darauf keinerlei oder nur begrenzt Einfluss. Sie können einen gewünschten Bereich angeben ohne eine Garantie dafür zu erhalten, dass die Schlüssel entsprechend generiert werden. Möglich wäre die Angabe von Schlüsselwörtern, die als Eingabe für die Generierung von Produktschlüsseln dienen. Somit könnte eine Abbildung von einem Schlüsselwort zu einem oder mehreren Produktschlüssel/n hergestellt werden.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Weiter oben wurde bereits angemerkt, dass das verwendete Tool bei der Umwandlung nicht nur auf eine Programmiersprache und Plattform abzielen darf. Dies wäre eine zu starke Einschränkung und würde die Attraktivität des PS herabsetzen. Andererseits ist es nicht notwendig, jede Plattform und Programmiersprache zu unterstützen. Dies ist wahrscheinlich auch gar nicht möglich. Daher müssen die gängigsten und verbreitetsten Technologien unterstützt werden. Darunter fallen Betriebssysteme wie Windows und unix-basierte Systeme, als auch die von Microsoft entwickelte .Net-Plattform und Suns Java Plattform. Sollte diese Unterstützung nicht durch ein einzelnes Tool gegeben sein, so müssten verschiedene zu Einsatz kommen.

Dieser Ansatz einer Dienstleistung ist als ESD-Erweiterung für das PS zurzeit keine Alternative. Der Einsatz eines inhaltsverändernden Werkzeugs, welches auch die Kontrolle über die Schutzmechanismen aus den Händen der Anbieter nimmt und auf das PS überträgt ist nicht

5.7 Sicherheitskritische Aspekte

Dieses Kapitel behandelt gesondert wichtige, die Sicherheit betreffenden Aspekte die bei der Entwicklung eines ESD-Systems im Kontext des PS eine Rolle spielen oder geachtet werden muss. Bei Softwareprodukten sind nicht nur die Registrierungsgeheimnisse zu schützen. Auch die Anbieter haben einige Einschränkungen und Kompromisse einzugehen.

Auch hier wird wie im Kapitel 5.4 nur auf den Ansatz aus Kapitel 5.3.1 eingegangen.

5.7.1 Übermittlung der Produktschlüssel und Produktschlüssellisten

Kommunikation zwischen PS und Anbieter-Systemen findet aus zwei Gründen statt. Zum einen werden Download-Anfragen für Produkte gestellt. Diese sind im öffentlichen Verzeichnis eines Anbieters und bedürfen keiner weiteren Schutzmechanismen.

Produktschlüssel und Produktschlüssellisten sind sensible Daten, die nur dem Anbieter bekannt sind. Erst bei der Registrierung beim PS wird durch die Übermittlung einer Notfallliste diesem ebenfalls ein Teil der Registrierungsgeheimnisse bekannt gemacht. Als letztes erhält ein Käufer für jede erfolgreiche Bezahlung einen Produktschlüssel.

Schon jetzt wird jeder Anbieter verpflichtet, in dem Verzeichnis mit dem zu schützenden Inhalt eine *.htaccess* Datei zu erstellen, welche durch die Angabe PS-spezifischer IP-Adressen diesem als einzigem System Zugriff zu gewähren. *.htaccess* sind Konfigurations-Dateien, die auf das sie enthaltene Verzeichnis Einfluss haben. Sie wurden mit der Entwicklung des Apache HTTP-Servers eingeführt und gehören standardmäßig zum Konfigurationsmanagement. Dieser Dateityp ist äußerst vielseitig einsetzbar. Die Apache Software Foundation rät allerdings von der Verwendung ab, falls der verwendete Server dem Nutzer gehört und er administrative Kontrolle besitzt. Da dies allerdings nicht als gegeben vorausgesetzt werden darf,

Konzeption einer Erweiterung zur elektronischen Software-Distribution

müssen Anbieter für den Schutz der Inhalte solch eine *.htaccess* nutzen bzw. ein Äquivalent, falls der verwendete Server einen anderen Mechanismus verwendet.

Angefragte Produktschlüssel sollten bei der Übertragung vor unbefugten Zugriffen geschützt werden. Dies kann durch die Nutzung einer gesicherten Verbindung über ein standardisiertes Protokoll wie SSL (Secure Socket Layer (46)) bzw. TLS (Transport Layer Security (47)) erreicht werden. Dieses Verfahren ist zu empfehlen, da die meisten Web-Server diese Sicherheitsprotokolle unterstützen, z.B. der Apache HTTP-Server, Microsofts IIS (Internet Information Service), etc. Jedoch ist auch hier wiederum die Frage, ob alle Anbieter ihre Systeme entsprechend konfiguriert haben bzw. ob sie die administrative Kontrolle besitzen, um für diese Schutzmechanismen die nötigen Voraussetzungen zu erfüllen. Sollte dies der Fall sein, sollten die Anbieter ihre Systeme entsprechend konfigurieren. Falls nicht, sollten Anbieter bei ihrem Webhoster nachfragen, ob sichere Verbindungen mittels SSL oder TLS möglich sind und zu welchen Konditionen. Da das PS bei jeder Schlüssel- bzw. Listenanfrage in die Rolle eines Client einnimmt, sollte bei der Verwendung von SSL oder TLS jeder Anbieter ein Sicherheitszertifikat erstellen. Dies kann von einer öffentlichen und beglaubigten Zertifikatsstelle (*Certificate Authority*) oder vom Anbieter selbst erstellt werden²¹. Obwohl die Authentifizierung mittels Zertifikaten als optional angeboten wird, ist sie zumindest serverseitig der Regelfall und bietet einen erhöhten Schutz bzgl. Authentifizierung der beteiligten Akteure als auch gegen so genannte „Man-in-the-Middle“-Angriffe. Die Beglaubigung eines Zertifikats kann nicht als gegeben angesehen werden. Dann sollten Anbieter ihre Zertifikate dem PS vor Veröffentlichung ihrer Produkte bekannt machen, z.B. während der Registrierung des ersten Produkts. Andererseits ist durch die Akzeptanz selbst erstellter Zertifikate die Authentifizierung als nicht sicher anzusehen. Somit besteht die größte Bedrohung in den schon erwähnten „Man-in-the-Middle“-Angriffen.

Sollte aus irgendeinem Grund keine Unterstützung der Sicherheitsprotokolle möglich sein, gäbe es die Alternative, durch Verwendung eines externen Kryptographie-Werkzeugs eine Art pseudo-sichere Kommunikation zu ermöglichen. Das Tool sollte moderne Verfahren zur symmetrischen Ver- und Entschlüsselung wie AES und DES oder public-key Verfahren wie RSA und OpenPGP unterstützen. Bei der Verwendung symmetrischer Verfahren muss dem PS der benutzte Algorithmus bekannt gemacht werden, als auch der entsprechende geheime Schlüssel, um die erhaltenen Daten zu entschlüsseln. Da es sich hierbei um sensible Daten handelt, wird jedem Anbieter geraten, den Schlüssel während der Produktregistrierung zu übermitteln.

²¹ Ein gültiges Format ist das in dem RFC 2459 beschriebene X.509

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Zu diesem Zeitpunkt wird eine sichere Verbindung über HTTPS durch den verwendeten Browser zum PS genutzt. Zum einen wird die Notfallliste übertragen, welche vor der Produktregistrierung in chiffrierter Form vorliegen sollte. Die ESD-Erweiterung lädt die verschlüsselte Datei hoch und benutzt den übergebenen Schlüssel, um die Liste in Klartext verwenden zu können. Sollte ein public-key Verfahren benutzt werden, so verschlüsselt der Anbieter seine Notfallliste mit dem öffentlichen Schlüssel des PS und stellt sie anschließend zum Upload zur Verfügung. ESD-intern wird der private Schlüssel zur Entschlüsselung genutzt. Für die Variante, in der das PS komplett für die Auslieferung verantwortlich ist, endet hier der Ablauf. Anbieter haben nur dafür Sorge zu tragen, dass immer eine weitere chiffrierte Datei vorhanden ist.

Der zweite Fall ist die Auslieferung eines einzelnen Produktschlüssels. Die Liste auf Seiten des Anbieters liegt in unverschlüsselter Form vor. Die Frage ist, wie man einem Käufer einen gültigen Schlüssel ausliefert, ohne ihn in Klartext zu übertragen? Es wäre ein unzumutbar hoher Aufwand, die zu verwendende Liste erst zu chiffrieren, anschließend an die ESD-Erweiterung des PS zu übertragen, welche sie intern entschlüsselt und einen Eintrag entnimmt. Da die Liste nur für ein einziges Datum benötigt wird, muss sie wieder chiffriert und an den Anbieter übertragen werden. Eine erneute Entschlüsselung ist nicht nötig, da die Liste bei jedem Kauf erneut an das PS übertragen werden muss. Bis auf den letzten Teil entspricht dies dem vorher beschriebenen Ablauf mit einer Notfallliste.

Zusätzlich finden die ressourcenbelastenden Kryptographie-Operationen hauptsächlich auf Seiten des PS statt, was zu einer extremen Belastung des Systems führen kann. Die Performance bei nebenläufiger Beantwortung weiterer Konsumentenfragen, die auch andere Waren betreffen, kann somit stark beeinträchtigt werden. Stattdessen sollten die Anbieterserver mehr einbezogen werden. Stellt das PS eine Produktschlüsselanfrage, sollte auch nur ein Schlüssel übertragen werden. Dieser wird aus der Liste entnommen und verschlüsselt übertragen. Da kryptographische Werkzeuge nur ganze Dateien verschlüsseln, muss der Produktschlüssel als einzelner Eintrag in eine temporäre Datei geschrieben und verschlüsselt übertragen werden. Die Weiterverarbeitung geschieht danach wiederum durch das PS.

Diese Lösungsmöglichkeit ist bei weitem nicht so sicher wie bei der Verwendung von SSL bzw. TLS. Es fehlen sämtliche Authentifizierungsmechanismen. Daneben ergibt sich das Problem, ein Kryptographie-Werkzeug mit den geforderten Eigenschaften zu finden und nutzen zu können. Zum einen muss es bei symmetrischen Verfahren die verwendeten Schlüssel exportieren können.

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Da es sich hierbei um das wohlgehütete Geheimnis handelt, gibt kaum ein Programm dieses selbst für den es benutzenden Entwickler preis²². Zum anderen müssen bei einer Kaufanfrage automatisch die beschriebenen Aktionen ausgeführt werden können. Genau dieser letzte Punkt ist wohl am schwierigsten zu realisieren. Beide Probleme sind wohl nur durch die Verwendung eines eigen- oder weiterentwickelten *Open Source* Werkzeugs zu lösen. Erschwerend ist auch die Tatsache, dass verschiedene Tools unterschiedliche Informationen während des Chiffriervorgangs hinzufügen. Somit ist die verschlüsselte Repräsentation immer unterschiedlich, sowohl inhaltlich als auch die codierte Form betreffend. Allein um diesen Sachverhalt vorzubeugen, müssen Anbieter und PS das gleiche Tool benutzen, welches konsequenterweise vom PS vorgegeben würde. Um allen Anbietern gerecht zu werden, muss das Tool als Mindestanforderung betriebssystem-unabhängig sein bzw. auf verschiedenen Betriebssystemen lauffähig sein. Ähnliches gilt für Werkzeuge mit asymmetrischen Verfahren. Und wieder steht die Frage der administrativen Kontrolle des Servers im Raum. Sollte ein Webhoster die Verwendung ausführbarer Programme auf seinen Servern untersagen, so gäbe es keine Möglichkeit Produktschlüssel chiffriert zu übermitteln. Das stellt ein relativ großes Sicherheitsrisiko gegenüber Lauschangriffen dar, von dem jedem Anbieter nachdrücklich abgeraten wird. Allerdings gilt es hier, die Intention, Einsatzumgebung und Nutzer des PS mit in die Überlegungen mit einzubeziehen. Das PS richtet sich an Einzelpersonen und kleinere bis mittlere Unternehmen. Die zum Erwerb angebotenen Software-Produkte werden aller Wahrscheinlichkeit nach nicht mehrere Hundert Euro kosten. Vielmehr sollen Entwickler kleinerer Programme ermutigt werden, ihre Produkte mittels *Super-Distribution-Channels* einer größeren Menge an potentiellen Nutzern bekannt zu machen. Es stellt sich daher die Frage, ob der Aufwand gesicherter Kommunikation tatsächlich dem Nutzen und auch der Vergütung lohnenswert ist.

5.7.2 Produktschlüssel und Einhaltung der Lizenzen

Die Philosophie des verwendeten Multilevel-Affiliated-Marketing Geschäftsmodells des PS mit Hauptausrichtung auf elektronische Musikdistribution basiert auf den Prinzipien des fairen Umgangs und Vertrauens zwischen Konsumenten und Anbietern. Um dieses zu unterstützen wird ein zusätzlicher Anreiz für alle Konsumenten geschaffen, am Vertrieb teil zu haben. Jede legal erworbene Musikdatei ist frei von Restriktionsmechanismen und kann jederzeit erneut über das PS heruntergeladen werden.

²² Dem Autor ist bis zum jetzigen Zeitpunkt kein Tool bekannt, welches den verwendeten symmetrischen Schlüssel auf Anfrage des Benutzers liefert

Konzeption einer Erweiterung zur elektronischen Software-Distribution

Sollte dies beim Kauf eines Produktschlüssels eines Softwareprodukts ebenfalls so gehandhabt werden? Das PS nimmt immer die Rolle eines Vermittlersystems mit zusätzlichen Dienstleistungen am Kunden ein. Gegenüber den Konsumenten besteht die Pflicht, jede erfolgreiche Transaktion auf der persönlichen Transaktionsseite anzuzeigen. Dazu zählen ebenfalls die erworbenen Produktschlüssel eines Softwareprogramms. Der Einwand, dass somit Softwarepiraterie begünstigt würde ist ebenso berechtigt wie auch zweifelhaft. Jede im Handel erhältliche Software die einen Registrierungscode verwendet und dieser auf irgendeine Weise dem Produkt beigelegt ist entspricht demselben Prinzip. Auch hier können mit nur einer legal erworbenen Lizenz mehrere Kopien auf mehreren Computersystemen installiert werden. Um Anbietern entgegen zu kommen, erhalten sie vom PS die Möglichkeit, die Anzahl unterschiedlicher Produktschlüsselkäufe zu begrenzen. Darüber hinaus ist es jedem Anbieter überlassen, seine Produkte mittels Online-Registrierungs- und/oder Aktivierungsmechanismen zu versehen und somit diesbezüglich geforderte Lizenzbedingungen auf korrekte Anwendung durch Konsumenten zu überprüfen. Dieser Vorgang liegt außerhalb des Verantwortungsbereichs des PS, da es eine direkte Kommunikation zwischen Anbieter und Konsument erfordert. Wie eine Registrierung im Detail angefertigt ist, bleibt jedem Anbieter selbst überlassen. Obwohl nicht wissenschaftlich nachgewiesen, werden zwanglose Registrierungen vermutlich von Konsumenten eher angenommen, da sie dadurch nicht in ihrer Entscheidungsfreiheit eingeschränkt werden. Zwanglose Registrierungen sind zum einen optional und die angegebenen Daten müssen nicht der Realität entsprechen. Zum anderen wird jeder Konsument, der Support, Informationen zu Updates und evtl. weitere angebotene Dienstleistungen eines Anbieters wünscht, bei legal erworbener Software seine Daten korrekt hinterlegen. Ein weiterer Grund ist die evtl. erhöhte Verbreitung eines Produkts. Angenommen es existieren 100 installierte Kopien, von denen zehn legal erworben und bezahlt, die verbliebenen 90 die Lizenzvereinbarungen verletzen und nicht vergütet wurden. Solch ein Szenario ist gegenüber einem, bei dem ein hervorragender Kopierschutz verwendet wird und nur die zehn legalen Kopien wieder zu finden sind, evtl. vorzuziehen. Sie sind vom finanziellen Standpunkt her gleich gut. Die Wahrscheinlichkeit, einen illegalen Nutzer durch das Produkt selbst zu überzeugen, der daraufhin die Möglichkeit zur legalen Nutzung und Support wahrnehmen möchte, kann somit erhöht werden. Verweist der Anbieter auf seiner Homepage auf den Vertrieb über das PS, so erfährt der potentielle Käufer über seine ihm vielleicht noch nicht bekannte einnehmbare Rolle als Wiederverkäufer, was ein weiterer Anreiz für den Erwerb einer legalen Kopie sein könnte.

6 Realisierung der Erweiterung

In diesem Kapitel wird auf die verwendeten Technologien eingegangen, auf denen das bestehende PS basiert. Ausgehend von den Überlegungen im vorherigen Kapitel 5 wurde das Auslieferungskonzept mit Notfallliste ausgewählt und realisiert.

6.1 Aktuelle Konfiguration des PotatoSystems

Die bisherige Realisierung des PS verwendet eine Vielzahl vorhandener Technologien, um die beschriebenen Funktionalitäten und Dienstleistungen zu realisieren. Dabei handelt es sich hauptsächlich um frei erhältliche *Open Source* Systeme. Die folgende Abbildung 9 zeigt die verwendeten Technologien und ihr Zusammenspiel. Zur Darstellung wird ebenfalls FMC genutzt, allerdings auf einem niedrigeren Abstraktionsniveau als in Abbildung 4.

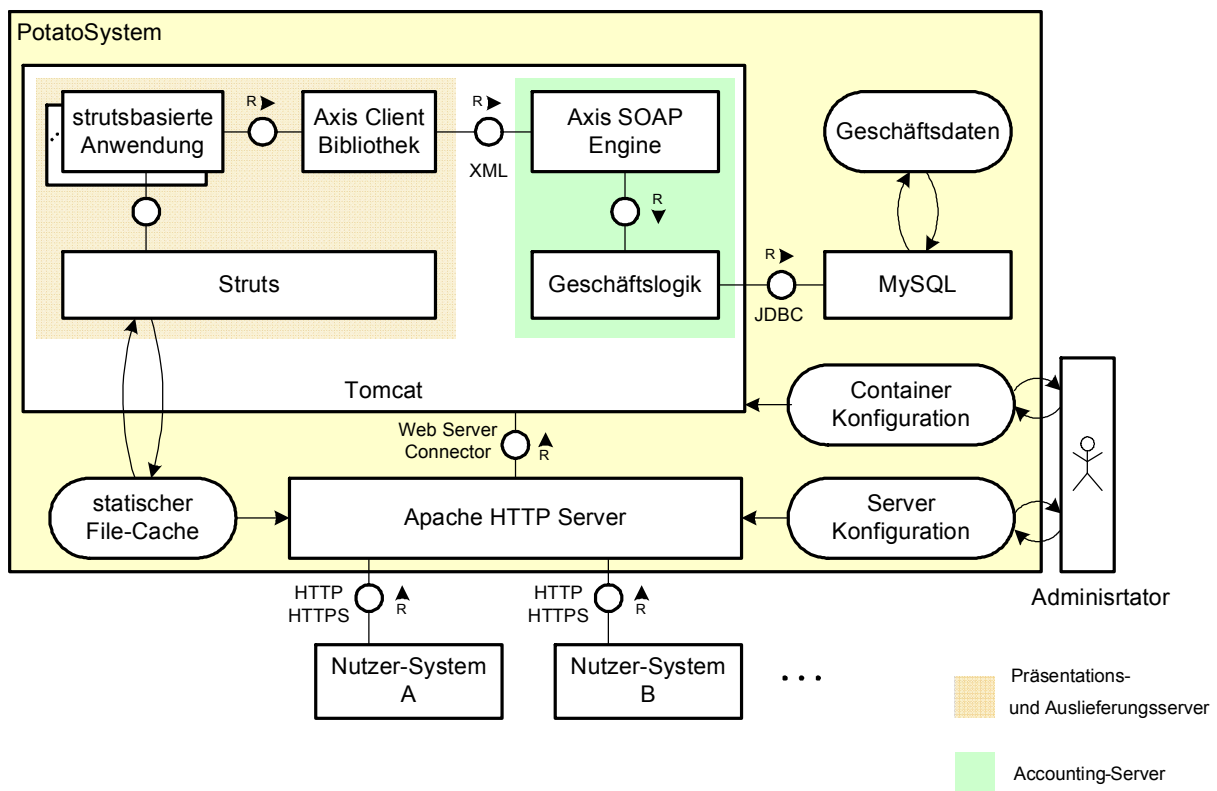


Abbildung 9 Verwendete Technologien und Systeme des PotatoSystems

Die Basis bildet ein Apache HTTP-Server. Dieser nimmt sämtliche Anfragen entgegen und beantwortet sicherheitsunkritische Anfragen, die informativen Charakter besitzen, bspw. die PS-Startseite. Erweitert wird der Apache durch das *Open Source* Projekt Tomcat. Es handelt sich um die Referenzimplementation einer Servlet/JSP-Engine von Sun Microsystems. Servlets und JavaServer Pages (JSP) sind eine Erfindung von Sun, um dynamischen Web-Content im Internet generieren zu können. Tomcat wird als Container bezeichnet, da er alle wichtigen

Realisierung der Erweiterung

Verwaltungsmechanismen ausführt. Es können mehrere Applikationen in nur einem einzigen Container laufen. Tomcat wurde in der Programmiersprache Java geschrieben, was zur Folge hat, dass alle weiteren verwendeten Systeme und angebotenen Dienstleistungen in Java implementiert sind. Zusammen ergibt der Apache HTTP-Server und Tomcat ein sehr mächtiges und funktionsreiches Konstrukt für die Realisierung von Web-Applikationen.

Das Dispatcher-System des Jacket (siehe Abbildung 4) basiert auf einem weiteren *Open Source* Projekt namens Struts. Dabei handelt es sich um ein das Model-View-Controller 2 (MVC2) Muster implementierendes aktionsgesteuertes Framework. Es nutzt Servlets und JSPs und eignet sich hervorragend, um Web-Applikationen einfach, komfortabel und schnell zu realisieren und im Tomcat-Container laufen zu lassen. Neben dem Jacket basiert ebenfalls der iFrame-Server auf Struts.

Der Accounting-Server (AS) des PS ist als Web-Service realisiert. Die Kommunikation zwischen AS, Jacket und iFrame-Server findet mittels SOAP (Simple Object Access Protocol) statt (siehe Abbildung 4). Als Protokoll-Realisierung wurde die in Java geschriebene, frei verfügbare Axis-Implementation gewählt.

Die vom AS verwendeten und berechneten Geschäftsdaten werden in verschiedenen Datenbanktabellen mittels des relationalen Datenbank-Management-Systems (DBMS) *MySQL* verwaltet. Die Kommunikation findet dabei über eine spezielle *Java Database Connectivity* (JDBC) Schnittstelle statt.

Die PS-spezifische Teilsysteme Accounting-Server, Jacket und iFrame-Server laufen zur Zeit in einem einzelnen Tomcat-Container. Der Einsatz der SOAP-Engine Axis ermöglicht eine Trennung der Subsysteme, um sie auf verschiedenen Servern innerhalb weiterer Servlet-Container laufen lassen zu können. Zusätzlich wird Anbietern, die sich mit einem Label- oder Portalaccount registriert haben eine Web-Service-Schnittstelle angeboten. Diese bietet Funktionalitäten zur Offline-Verwaltung der registrierten Waren, Massenregistrierung und die Nutzung des PS als Application Service Provider (letzteres nur für Portalbetreiber) an.

6.2 Prototypische Realisierung

Basierend auf den vorhandenen Strukturen und Abläufen des PS werden in diesem Kapitel exemplarisch verschiedenste Ausschnitte der Realisierung der ESD-Erweiterung, wie im Kapitel 5.3.1 beschrieben, gezeigt.

6.2.1 Das ESD-Modul im PotatoSystem

Das Konzept aus Kapitel 5.3.1 mit der Verwendung einer Notfallliste, welches in Kapitel 5.4 genauer beschrieben wird, ist als ein eigenes Subsystem realisiert.

Realisierung der Erweiterung

Es kann komplett losgelöst vom PS existieren und als eigenständiger Dienst in Verbindung mit dem auf Anbieterseite notwendigen Skript laufen. Realisiert werden musste die Verwaltung und Auslieferung der Produktschlüssel und Notfalllisten, als auch Maßnahmen, die ein Produkt aus dem Verkauf ausschließen bzw. wieder zulassen. Die Shareware wird durch schon vorhandene Abläufe im bestehenden PS ausgeliefert. Die ESD-Erweiterung auf Seiten des PS ist aus Kompatibilitätsgründen in der Programmiersprache Java geschrieben. Dazu wurde das von Sun Microsystems frei erhältliche *Java 2 Standard Edition Development Kit, Version 1.4.3_04* (j2sdk1.4.3_04) verwendet. Es werden nur die von der *Java Runtime* zur Verfügung gestellten Funktionalitäten genutzt.

Die gewünschte Funktionalität wird modular gekapselt in zwei Klassen namens *PESD* und *EmergencyListCatcher* verwirklicht. Die folgende Abbildung 10 zeigt die statische Struktur als UML-Klassendiagramm. Es werden neben dem Klassennamen nur die Methoden modelliert, da die Liste der zugehörigen Variablen das Modell unnötig vergrößern und in keiner Weise zu einem besseren Verständnis oder sinnvoll zu interpretierenden Informationsgehalt in sich tragen würde.

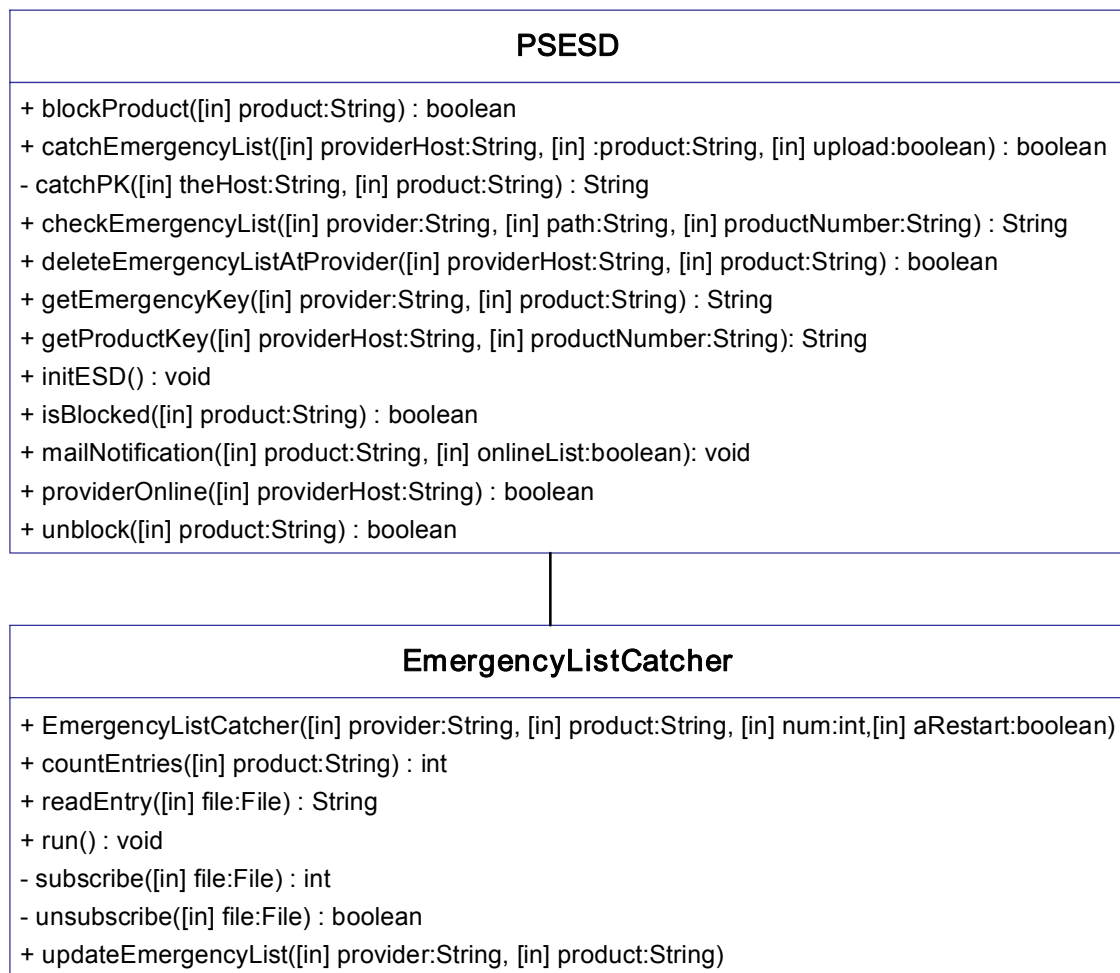


Abbildung 10 UML Klassendiagramm der realisierten ESD-Erweiterung

Realisierung der Erweiterung

Bis auf zwei Ausnahmen bieten beide Klassen nur statische Klassen-Methoden zur Durchführung der gewünschten Funktionalität an. Dies erhöht die Performance der Beantwortung einer Anfrage durch den Proxy, da die Instanziierung eines Objekts entfällt. Zudem werden Ressourcen nur im Falle eines benötigten Einsatzes angefordert.

Die Klasse *PSESD* bietet ausschließlich statische Methoden an. Es existiert kein öffentlicher Konstruktor, mittels dessen ein Objekt instanziiert werden könnte. Sie bildet den Hauptteil des Subsystems und realisiert alle Abläufe, die zur Verwaltung der Produktschlüssellisten und zur Auslieferung einzelner Produktschlüssel konzipiert wurden.

Einstiegspunkt bildet die Methode *initESD()*, durch die zwei Verzeichnisse angelegt werden:

- *esdEmergencyLists*, in dem alle Notfalllisten als separate Textdatei hinterlegt werden
- *blocked*, enthält eine Textdatei mit vom Verkauf ausgeschlossener Produkte

Sind beide Verzeichnisse angelegt, wird anschließend überprüft, ob eine Datei namens „catcher.txt“ existiert bzw. ob die Datei Einträge besitzt. Diese Datei wird von der Klasse *EmergencyListCatcher* angelegt und wird in diesem Kapitel weiter hinten beschrieben.

Die Methode *checkEmergencyList(...)* wird bei jedem Zugriff auf eine Notfallliste aufgerufen. So auch nach jeder erfolgreichen Produktregistrierung. Sie prüft zuerst die Existenz einer Notfallliste. Sollte die zu einem Produkt gehörende Liste nicht im Verzeichnis *esdEmergencyLists* vorhanden sein, wird ein neuer Prozess gestartet, in dem ein Objekt der Klasse *EmergencyListCatcher* instanziiert wird. Existiert eine Liste wird sie zusätzlich inhaltlich überprüft. Wird festgestellt, dass eine Liste nicht existiert oder die Anzahl der Einträge unter 20 fällt, wird dies dem Anbieter mittels einer E-Mail mitgeteilt. Der Versandt wird mit dem Aufruf der Methode *mailNotification(...)* durchgeführt. Sie wird ebenfalls aufgerufen, wenn durch die Methode *getProductKey(...)* festgestellt wurde, dass die Onlineliste des Anbieters nur noch 19 Einträge enthält. *getProductKey(...)* ist für die Kaufabwicklung die wichtigste Methode. Sie wird bei einer Kaufanfrage aufgerufen und baut eine einfache HTTP-Verbindung zum Anbieter-Server auf, deren Endpunkt ein Skript im geschützten Verzeichnis ist²³. Zusätzlich werden bei dieser Verbindung zwei Argumente übergeben. Das erste besagt, dass es sich um die Auslieferung eines Produktschlüssels handelt. Das zweite Argument ist die das Produkt eindeutig identifizierende Kennung.

Sollte bei der Kommunikation ein Fehlverhalten auftreten, z.B. das keine Verbindung zum Anbieterserver hergestellt werden konnte, wird überprüft, ob der Anbieter erreichbar ist.

²³ Dieses Skript wird im folgenden Kapitel 6.2.2 beschrieben.

Realisierung der Erweiterung

Anstelle eines Ping-Signals wie in Kapitel 5.4.2.1 beschrieben, wird versucht das Skript zu erreichen, welches in diesem Fall die Rolle eines einfachen Echo-Dienstes²⁴ übernimmt. Ein Ping-Signal kann nur feststellen, ob ein Netzwerkknoten verfügbar ist. Wird das Skript als Echo-Dienst verwendet, kann zusätzlich die Konnektivität zum geschützten Verzeichnis überprüft werden. Dies ist vor allem dann zweckmäßig, wenn der Anbieter bei einem Drittanbieter hostet. So kann der Fall auftreten, dass der Server des Web-Hosters verfügbar ist, der Anbieter aber nicht²⁵. Dem Skript wird eine Zeichenkette übermittelt, die, falls empfangen, umgehend zurückgeschickt wird. Die vorgeschlagenen Zeitintervalle können übernommen werden. Sollte dieser Versuch scheitern, wird für die Beantwortung der Anfrage auf die Notfallliste zurückgegriffen. Andernfalls wird eine erneute Schlüsselanfrage gestellt. Sollte diese aus irgendeinem Grund ebenfalls scheitern, wird wiederum ein Eintrag aus der Notfallliste mittels *getEmergencyKey(...)* entnommen.

Unabhängig woher, liefert die Methode *getProductKey(...)* einen gültigen Produktschlüssel oder eine leere Zeichenkette zurück. Letzteres ist der Fall, falls der Anbieter keinen Produktschlüssel liefern kann²⁶ und die bereitgestellte Notfallliste keinen Eintrag mehr enthält. Gleichzeitig wird das Produkt in die Liste geblockter Produkte im Verzeichnis *blocked* mittels der Methode *blockProduct(...)* aufgenommen. Die Liste wird in der Datei „blockedList.txt“ persistent gespeichert. Erst wenn der Anbieter wieder eine Notfallliste mit wenigstens einem Eintrag erfolgreich an das PS ausliefern konnte, wird der Eintrag des entsprechenden Produkts aus der Liste mittels der Methode *unblock(...)* entfernt. Zuständig dafür ist ein Objekt der Klasse *EmergencyListCatcher*. Diese instanziierbare Klasse wird von *PESD* jedes Mal aufgerufen und als separater Prozess gestartet, wenn für ein Produkt die Anzahl vorhandener Einträge in der Notfallliste unter den willkürlich festgelegten Wert 20 sinkt. Für jedes Produkt existiert immer nur ein solcher Prozess. Das bedeutet, dass die maximale Anzahl an gestarteten Prozessen der Klasse *EmergencyListCatcher* gleich der Anzahl registrierter Softwareprodukte sein kann. Um dies zu Garantieren wird eine einfache Textdatei namens „catcher.txt“ vom ersten jemals auf diese Weise erzeugten Prozess erstellt. Die Kennung des Produkts als auch die Host-Adresse des Anbieters werden dem Objekt bei seiner Initialisierung als Parameter mit übergeben und mittels der Methode *subscribe(...)* in die Liste bestehender „Notfalllistenhäscher“ eingeschrieben.

²⁴ IANA definiert Echo als Standard-Dienst auf Port 7, der i.A. von allen *nix-Derivaten implementiert wird. Windowssysteme besitzen diesen Dienst nicht.

²⁵ Bspw. hat der Anbieter seine vertraglichen Leistungen mit dem Web-Hoster nicht erfüllt und ist gesperrt worden

²⁶ Kommunikations- oder Serverfehler oder der Anbieter hat keinen Produktschlüssel in seiner Online-Liste auf dem Web/File-Server

Realisierung der Erweiterung

Jeder neu gestarteter Prozess überprüft nach seiner erfolgreichen Initialisierung selbstständig, ob bereits ein Eintrag vorhanden ist. Falls dies der Fall sein sollte, wird der Prozess sofort terminiert. Ein weiterer Verwendungszweck der Liste innerhalb der Datei „catcher.txt“ zeigt sich nach einem Neustart des PS. Für jeden Eintrag muss erneut ein Prozess gestartet werden.

Die Hauptaufgabe besteht im Holen einer Notfallliste vom Anbieter. Dazu wird die Methode *catchEmergencyList(...)* aufgerufen. Diese wird von der Klasse *PSESD* bereitgestellt, da nur sie für alle Verwaltungsaufgaben bzgl. Produktschlüssellisten vorgesehen ist. Sollte ein Uploadversuch misslingen, wird der Prozess für ein bestimmtes Zeitintervall „schlafen gelegt“, d.h. er behält zwar weiterhin seine ihm zugewiesenen Speicherressourcen, verbraucht aber keinerlei CPU-Zeit. Kann jedoch eine neue Liste geholt werden und ist die Anzahl der Einträge der Notfallliste größer als 20, wird der Produkteintrag aus der Datei „catcher.txt“ entfernt. Sollte das Produkt geblockt sein, wird es mittels *unblock(...)* wieder für den Verkauf freigegeben. Abschließend terminiert der Prozess und gibt alle Systemressourcen wieder frei. Bleibt die Anzahl an Einträgen kleiner als 20, setzt das Objekt seine Aufgabe fort.

Die Klasse *EmergencyListCatcher* soll nur den einen Zweck erfüllen, in bestimmten Intervallen bei dem entsprechenden Anbieter nachzufragen, ob eine Notfallliste ins PS kopiert werden kann. Das Erzeugen eines separaten Prozesses begründet sich durch die bestehende Realisierung des PS. Der Apache HTTP-Server hat intern nach erfolgreichem Start einen so genannten Master-Server-Prozess zu laufen, der für die Erzeugung und Verwaltung so genannter Child-Server-Prozesse zuständig ist. Die initiale Anzahl ist in der Konfigurationsdatei *httpd.conf* festgelegt²⁷. Für jeden eingehenden Request wird ein so genannter Worker-Thread benutzt. Übersteigt die Anzahl eingehender Requests die Anzahl vorhandener Child-Prozesse bzw. verfügbarer Worker-Threads, werden weitere erzeugt. Solange eine Verbindung besteht, befinden sie sich in einer so genannten *request-response loop*. Diese wird erst verlassen, wenn die Verbindung geschlossen wird. Die verwendeten Worker werden entweder terminiert, um die initiale Anzahl wieder herzustellen oder für weitere Verbindungswünsche wiederverwendet. Eingehende Requests, die von dem Servlet-Container (in diesem Fall Tomcat) behandelt werden sollen, werden weitergeleitet. Auf Servlets basierende Web-Applikationen sind in ihrer logischen Struktur genauestens spezifiziert. Die den Request behandelnde Applikation basiert auf dem Struts-Framework und erweitert es mit anwendungsspezifischer Ausführungslogik.

²⁷ Dies ist abhängig vom verwendeten Betriebssystem. Windows-Systeme verwenden nur einen einzigen Child-Prozess, der eine festgelegte Anzahl an Threads startet. *nix-Systeme starten hingegen Prozesse, die intern weitere Threads zur Durchführung ihrer Aufgabe starten

Realisierung der Erweiterung

Gesetzt dem Fall, das die Voraussetzungen für die Instanziierung eines *EmergencyListCatcher*-Objekts gegeben sind, ist das Kreieren eines neuen, von Struts und Tomcat unabhängigen Prozesses zweckmäßig. Dieser existiert nach Beenden des Requests und wird bei einem evtl. Neustart des Tomcat-Containers nicht terminiert. Letzteres wäre bei einer Realisierung auf Thread-Basis nicht der Fall. Somit wird das Teilsystem unabhängig vom bestehenden System. Einzig ein kompletter Neustart des gesamten Serversystems würde die Prozesse beenden.

6.2.2 Das ESD-Skript im geschützten Anbieterverzeichnis

Das PS stellt jedem Anbieter ein Skript namens *keyRetrieverPS.php* zur Verfügung. Aus der Datei-Endung ist ersichtlich, dass es in der Sprache PHP (PHP Hypertext Processor) geschrieben ist. Damit es auf einem Anbieter-Server ausgeführt werden kann, muss der Server entsprechend konfiguriert sein. Es werden nur Funktionen und vordefinierte Variablen der Sprache verwendet, die seit der Version 4.1.0 im Repertoire zu finden sind. Es kann eingesehen und modifiziert werden, solange die externen Funktionalitäten nicht verändert werden.

Das Skript hat vier Aufgaben im Auslieferungszyklus der ESD zu erfüllen. Daher werden von der ESD-Erweiterung des PS verschiedene Parameter in vorgegebener Reihenfolge erwartet. Das erste Argument namens „do“ bezeichnet die zu vollziehende Aufgabe und kann folgende Werte haben:

- *daily*– liefere einen einzelnen Produktschlüssel
- *necessary*– Behandlung einer Notfallliste
- *test*– Überprüfung der Erreichbarkeit des Skripts
- *gratz*– Löschen der zuvor übertragenen Notfallliste

Das zweite Argument ist abhängig vom Wert des ersten und wird wie folgt abgebildet:

do-Wert	Argument 2	Wert	Argument 3	Wert
daily	esdPKFile	Produktidentifizierung		
necessary	esdel	Produktidentifizierung	upload	true false
test	psecho	Zeichenkette		
gratz	keyfile	Produktidentifizierung		

Tabelle 3 Abhängige Argumente für ESD-Skript auf Anbieterseite

Prinzipiell identifiziert das zweite Argument das verwendete Produkt. Einzige Ausnahme ist, wenn das erste Argument „do“ mit dem Wert „test“ belegt ist. Das folgende Argument „psecho“ enthält eine Zeichenkette, deren Wert von der ESD-Erweiterung des PS gesendet wird.

Realisierung der Erweiterung

Entspricht „do“ dem Wert „necessary“ existiert ein drittes Argument namens „upload“, welches festlegt, ob nur die Existenz einer Notfallliste geprüft werden soll („false“) oder sie an das PS übertragen werden soll („true“);

6.3 Von der Registrierung bis zum Verkauf

Bevor ein beim PS angemeldeter Anbieter seine Software vertreiben kann, muss er die in Kapitel 5.5 geforderten Voraussetzungen erfüllen, die jetzt konkretisiert werden:

- Das Produkt liegt als „Prüf-vor-Kauf“ Sharewareversion im öffentlichen Verzeichnis vor
- Es existieren zwei beliebig benannte Textdateien, wobei eine die Zeichenkette „_PSUpload“ in ihrem Namen trägt. Sie liegen im durch eine *.htaccess*-Datei geschützten Verzeichnis

Das Produkt kann daraufhin beim PS registriert werden. Ist dieser Prozess abgeschlossen, erhält der Anbieter das Skript, einen Verkaufslink und eine vom PS generierte Kennung.

Der Link enthält spezifische Daten und erlaubt es Konsumenten, für das Produkt einen Produktschlüssel zu erwerben. Daher sollte er auf der Website des Anbieters veröffentlicht werden. Eine zusätzliche Option wäre die Ersetzung der im Produkt bestehenden URL durch die im Verkaufslink. Den Käufern würde der Umweg über die Website des Anbieters erspart bleiben und sie wären sofort im Erwerbsprozess des PS. Allerdings müsste das Produkt nochmals durch den Shareware-Prozess. Es liegt im Ermessen des Anbieters, ob dieser Aufwand in Kauf genommen wird, um den Käufern den Erwerbsprozess zu erleichtern.

Die vom PS generierte Kennung wird zur Umbenennung der Produktschlüssel-Dateien im geschützten Verzeichnis genutzt, um sie eindeutig identifizieren zu können. So lange dies nicht geschehen ist, kann die Notfallliste nicht ins PS kopiert werden, was zur Folge hat, dass das Produkt noch nicht für den Verkauf freigegeben wird. Sind alle Vorbedingungen erfüllt, können Konsumenten die Software zum Test herunterladen. Besteht der Wunsch einen Produktschlüssel zu erwerben, kann über einer der bereitgestellten Möglichkeiten der Kaufvorgang angestoßen werden, wie in Kapitel 5.3.1 und Kapitel 6.2 beschrieben.

7 Schwächen der Realisierung und des verwendeten Konzepts

Die ESD-Erweiterung für das PS folgt konzeptionell und in der Realisierung Vorgaben der 4FO AG. Sie sollte minimal-invasiv sein, einfach ins bestehende System zu integrieren, für Nutzer eine simple Bedienung aufweisen und Anbieter nicht von ihren Verpflichtungen entbinden. Letzteres beinhaltet eine ständige Verfügbarkeit der Server und Waren der Anbieter. Diesen Vorgaben folgend entstand die in Kapitel 5 und Kapitel 6 beschriebene Realisierung eines lauffähigen Prototyps. An dieser Stelle sollen nun einige Schwachstellen aufgezeigt und Verbesserungen für weitere Entwicklungsschritte vorgeschlagen werden. Die meisten dieser Schwachstellen sind durch so genanntes *Refactoring* (54) zu beheben, andere benötigen eine konzeptionelle Überarbeitung. Unter *Refactoring* wird ein Prozess zur internen Neustrukturierung vorhandenen Codes verstanden. Dies ist bspw. dann nötig, wenn Systeme durch Weiterentwicklungen stark von ihrem ursprünglichen Design abweichen. Auf Prototypen basierende Entwicklungen können ebenfalls diese Art der Wiederverwendung bestehenden Codes verwendet werden. Wichtigste Bedingung ist das Beibehalten des externen Verhaltens refactorisierter Programmteile.

Die bestehende Implementierung realisiert ESD nach der ersten Definition aus Kapitel 2.1.1. Ein vollständiges System würde eine zusätzliche Erweiterung auf Konsumentenseite bedeuten. Der Ablauf würde dahingehend erweiterte, dass der Konsument nach erfolgreichem Download der Testversion den Start einer automatischen Installationsroutine angeboten bekommt. Es liegt in seinem Ermessen diese durchzuführen. Benötigt würde auf Konsumentenseite ein vom PS bereitgestelltes Subsystem, auf das nur das PS Zugriff hat bzw. das nur mit dem PS kommuniziert. Es würde die Rolle einer zentralen Verwaltungsinstanz aller über das PS erhaltenen Softwareprogramme einnehmen. Kopiert der Konsument eine Testversion auf sein System, wird dessen Speicherort dem Subsystem automatisch vom PS mitgeteilt und die entsprechende Installationsdatei aufgerufen. Eine andere Möglichkeit wäre, dass alle Downloads in ein vorher bestimmtes Verzeichnis kopiert würden, das dem Subsystem bekannt ist. Im einfachsten Fall wird ein Skript verwendet. Ein höher entwickeltes System wäre jedoch wünschenswert, um die Benutzerfreundlichkeit mittels einer GUI zu erhöhen, durch die das Subsystem vom Konsumenten administriert werden könnte.

Entschließt er sich einen gültigen Produktschlüssel zu erwerben, wäre eine automatische Freischaltung durch das ESD wünschenswert.

Schwächen der Realisierung und des verwendeten Konzepts

Dazu müsste jeder Anbieter dem PS mitteilen, wie der Mechanismus und die verwendeten Datentypen spezifiziert sind und die Möglichkeit zur automatischen Freischaltung zu gewährleisten. Dieser Aufwand wäre eine Dienstleistung am Kunden, benötigt aber eine entsprechende Administration als auch Verwaltung auf Seiten des PS. Diese Kapazitäten besitzt die 4FO AG nicht. Daher könnten solche Dienste nur angeboten werden, wenn das PS die Formate, Protokolle und Mechanismen bzw. Schnittstellen spezifizieren würde, an die sich die Anbieter halten müssen. Dies ist wiederum eine Einschränkung, die den Anbietern einen Teil ihrer Entscheidungsfreiheit nimmt und zudem nicht von allen unterstützt werden kann. Letzteres wäre durch das Bereitstellen eines Tools zur Generierung von Shareware zu umgehen. Die nötigen Datenstrukturen und Abläufe wären bekannt. Eine verbesserte automatisierte Kaufabwicklung wäre erzielt.

Die ESD-Erweiterung für das PS kann sich mit bestehenden kommerziellen Lösungen nicht vergleichen lassen. Ein Beispiel ist Firma *element5* (51), die als eCommerce-Outsourcing Partner den Online-Vertrieb der Produkte ihrer Kunden komplett übernehmen. Diesen Anspruch hat das PS nicht. Vielmehr geht es um die Bereitstellung eines Vertriebskanals zur Auslieferung virtueller Waren, in dem das PS die Rolle eines Vermittlers einnimmt. Anbieter sind ein fester und unverzichtbarer Bestandteil innerhalb der Wertschöpfungskette und werden daher auch stark mit eingebunden. Die daraus entstehenden Verpflichtungen sind unvermeidbar. Ihnen nachzukommen ist zwar kein unzumutbarer Aufwand, kann allerdings noch weiter erleichtert werden. Die Aufspaltung der generierten Produktschlüsselliste kann durch das ESD-Teilsystem erfolgen. Dazu würde eine bestimmte Anzahl an Einträgen, bspw. die ersten 50 in das PS transferiert und aus der Liste entnommen werden. Diese Notfallschlüssel würden in einem hoch entwickelten ESD nicht in einer separaten Textdatei gespeichert, sondern in einer für jedes Produkt separaten Datenbanktabelle. Die Funktionalitäten eines DBMS würden wesentliche Hilfestellungen bei der Verwaltung liefern, z.B. Statusabfragen durch den Anbieter, wie viele Schlüssel noch für den Notfall vorhanden sind, etc. Der Präsentations- und Auslieferungsserver (vgl. Abbildung 7) nutzt das vorhandene MySQL-DBMS intern zum loggen Struts-spezifischer Aufrufe, um im Falle eines Systemfehlers Debugging-Informationen zu erhalten. Das ESD-Teilsystem soll hingegen keinen Zugriff auf das DBMS haben. Dies liegt vor allem an dem zu erwartenden erhöhten Administrations- und Verwaltungsaufwand. Zusätzlich müssen Attacken auf das DBMS von vornherein abgewendet werden. So ist bspw. die Gefahr einer SQL-Injection-Attacke sehr hoch.

Schwächen der Realisierung und des verwendeten Konzepts

Böswillige Anbieter könnten versuchen, anstelle einer Notfallliste einen Befehl zum Löschen einer Datenbanktabelle zu senden oder Zugriff auf das System zu erlangen. Solche Szenarien müssen verhindert werden, indem das ESD-Teilsystem die kopierten Zeichenketten analysiert und gegebenenfalls verwirft.

Ein weiterer Vorteil der Verwendung eines DBMS besteht in der Möglichkeit, die bestehende Realisierung in ihrer statischen Struktur zu entkoppeln. Die verwendeten Verzeichnisstrukturen und Dateipfade verlangen, dass die der Klassen *PSESD* und *EmergencyListCatcher* zurzeit in ein und demselben Verzeichnis existieren. Würde die Hauptklasse des ESD-Teilsystems zusätzlich zu einer Web-Applikation erweitert werden, könnten die Vorteile und Features des Tomcat-Containers genutzt werden. Tomcat unterstützt RDBMS-Zugriffe via *Java Database Connectivity (JDBC)* und eröffnet somit die Möglichkeit, das ESD-Teilsystem durch einen einheitlichen und durch eine andere Instanz verwalteten Zugriffsmechanismus flexibler zu gestalten. Zudem könnte es verteilt in verschiedenen Container und somit verschiedenen Adressräumen laufen.

Eine gravierende Schwachstelle bilden aufgebaute Verbindungen zum Anbieterserver. Es handelt sich um einfache, unverschlüsselte HTTP-Verbindungen. Die übertragenden Daten sind in „*plaintext*“ lesbar. Lauschangriffen ist somit Tür und Tor geöffnet. Die in jedem mitgeschnittenen Datenstrom enthaltenen Informationen können sofort ausgelesen und interpretiert werden, ohne aufwendige Kryptoanalysen durchführen zu müssen. Ebenfalls inhaltsverändernde Angriffe sind uneingeschränkt möglich. In Kapitel 5.7.1 wird auf weitere Gefahren hingewiesen und die Verwendung standardisierter Kommunikationsprotokolle mit Verschlüsselungsmechanismen empfohlen. Das solche Schutzmechanismen nicht zum Einsatz kommen basiert auf der Philosophie des PS. Es muss jedem Nutzer, der die Rolle eines Anbieters einnimmt, ermöglicht werden, seine Waren zum Verkauf anbieten zu können. Daher müssen auch diejenigen berücksichtigt werden, die einen einfachen Account bei einem Drittanbieter besitzen und die verwendete Technologie nicht unterstützen oder für die es ein unzumutbar hoher finanzieller Aufwand wäre, solch eine Unterstützung zu erhalten. Genau diese ist eine der Zielgruppen des PS. Die zum Erwerb registrierten Programme werden aller Voraussicht nach keine teuren, hoch entwickelten Produkte sein und das Interesse für Hacker nur gering wecken. Daher wurde auf Verschlüsselungen verzichtet.

Wie unflexibel die bestehende Realisierung ist, zeigt die Verwendung fest encodierter Dateinamen und des festgelegten Skriptnamens auf Anbieterseite. Ist kein DBMS verwendbar, so kann durch die Nutzung einer graphischen Benutzerschnittstelle (GUI – *graphical user interface*) mittels manueller Pfadangabe ein wenig mehr Flexibilität erreicht werden.

Schwächen der Realisierung und des verwendeten Konzepts

In Kapitel 6.2.1 werden die genutzten Verzeichnisse und Dateipfade beschrieben. Über eine GUI könnten somit auf Seiten des PS einige wenige strukturelle Einstellungen vorgenommen werden, die eine Neuorganisation ermöglichen, ohne das ESD-Teilsystem neu kompilieren zu müssen. Dies gilt sowohl für die Ersteinrichtung als auch für den laufenden Betrieb.

Eine weitere Verbesserung könnte durch die Anpassung der Nachfragezeiten der Klasse *EmergencyListCatcher* erreicht werden. Für jedes Produkt, dessen Notfallliste weniger als 20 Einträge aufweist, wird ein separater Prozess der Klasse *EmergencyListCatcher* erschaffen. Ist die Anfrage beim Anbieter vergeblich gewesen, wird seine aktuelle Ausführung für den Zeitraum von einer Stunde unterbrochen. Würde eine weitere instanziiierbare Klasse existieren, die entweder periodisch alle Notfalllisten überprüft, ob sie verändert wurden oder nach einem Zugriff, könnte das Nachfrage-Intervall bei Bedarf modifiziert werden. Diese Klasse sollte als Singleton laufen, um möglichst wenig Ressourcen zu verbrauchen. Andererseits ist die Änderung nach Bedarf auch nur dann nötig, wenn die Notfallliste eines Produkts zur Neige geht und keine neuen Schlüssel vom Anbieter geliefert werden. Dies wird durch eine rechtzeitige Warnung per eMail verhindert. Einzig die Annahme, dass sich ein Produkt derart gut verkauft, dass sowohl Online- als auch Notfallliste innerhalb kürzester Zeit geleert wären, könnte diese Modifikation rechtfertigen.

Ein *EmergencyListCatcher* Objekt wird für jedes Produkt einzeln gestartet. Im schlimmsten unzunehmenden Fall würden so viele Prozesse wie registrierte Softwareprodukte gestartet. Dieser Fall ist allerdings nur anzunehmen, wenn die physischen Verbindungen zwischen PS und den Anbietern aus irgendeinem Grund fehlerhaft sind. Hingegen ist ein Szenario, in dem ein Anbieter viele Produkte registriert hat und nicht erreichbar ist wahrscheinlicher. Für jedes dieser Produkte potentiell einen neuen Prozess zu starten ist durchaus zweckmäßig. Der Anbieter muss nicht alle Produkte auf demselben Server zu liegen haben. Es würden nur für Programme Anfragen gestellt, die sich auf dem ausgefallenen Server befinden.

Die Registrierung einer virtuellen Ware umfasst die Generierung eines Hash-Wertes, durch den das Produkt eindeutig identifiziert werden kann. Bei der Musikdistribution wird dieser Wert herangezogen, um bei der Auslieferung einer erworbenen Ware deren Korrektheit zu gewährleisten. Der Download von Vorhördateien wird hingegen nicht protokolliert und erfolgt anonym. Da sowohl Vorhördateien als auch Shareware-Programme im öffentlichen, ungeschützten Verzeichnis eines Anbieters liegen, wird der Hash zu keiner Zeit verwendet. Das bedeutet, dass Anbieter jederzeit ihre Waren austauschen oder modifizieren könnten, ohne Überprüfung durch das PS befürchten zu müssen.

Schwächen der Realisierung und des verwendeten Konzepts

Es sollte überlegt werden, Softwareprodukte ebenfalls zu cachen, um Konsumenten nur die ihnen angebotenen Produkte ausliefern zu können und sie somit vor evtl. Schaden zu bewahren. Bspw. könnte ein Anbieter ein Produkt zur Desktopverwaltung registrieren, das später mit einem Trojaner erweitert wird, um die Konsumentenrechner auszuspionieren.

8 Zusammenfassung

Elektronische Software Distribution ist als die zurzeit innovativste Form der Verteilung von Software zu betrachten. Die Möglichkeit der Verteilung einer rein virtuellen Ware über technische Kommunikationskanäle und die dazu gehörenden Dokumentationen ebenfalls als virtuelles Gut ausliefern zu können, unterstützt die Vision einer papierfreien Arbeitsumgebung. Die veralteten Distributionskanäle über CD oder DVD werden mit der Zeit immer mehr an Bedeutung verlieren. Viele Unternehmen und Entwickler haben bereits den Grundbaustein für ESD-Systeme gelegt, indem sie den Auslieferungsteil bereits kommerziell betreiben. Sie bieten ihrer Waren online an und ermöglichen den Download von Testversionen. Der Kauf einer Lizenz schaltet das Produkt frei, so dass es zu einer Vollversion wird. Zusätzlich erhält der Kunde Service- und Supportleistungen. Jedes Unternehmen muss für sich selbst die Entscheidung treffen, ob sich der Einsatz eines eigenen ESD-Systems finanziell rechnet und ob der Administrations- und Verwaltungsaufwand im Verhältnis zum Nutzen stehen. Alternativ existiert die Möglichkeit des Outsourcing. Ein vollständiges ESD wird vorerst nur zwischen Unternehmen durchgeführt werden, die in vertraglicher Kooperation miteinander stehen. Für den Einsatz beim Endkunden ist nur ansatzweise zu denken. Denn die wenigsten Nutzer möchten eine Installation durch fremde Hände auf ihren Heimrechnern initialisiert haben. Genauso wenig wie ein unbefugtes und nicht gewolltes Ausführen von Programmen auf ihrem Rechner, wünschen sich Konsumenten durch ihre Angaben bei Registrierungen durch Marketingabteilungen großer Firmen manipuliert oder mit Werbe-Mails überschüttet zu werden. Schlimmer noch wäre die Verletzung der Geheimhaltungspflicht persönlicher Daten. Firmen sollten sich daher bemühen ihre Kunden mit Respekt zu behandeln und die von ihnen gewünschten Hilfestellungen und Informationsangebote zu teil werden zu lassen.

Neben der technischen Realisierung ist das große Thema der Schutz der Waren vor unbefugter, gegen die Lizenz verstoßender Nutzung. Softwarepiraterie stellt ein für die Existenz vieler kleiner und mittlerer Unternehmen, die auf die Rückführung ihrer Investitionen stärker angewiesen sind als große Firmen mit finanziellen Rücklagen, großes Problem dar. Offizielle Statistiken zeigen, dass Softwarepiraterie und die Nutzung solcher nicht lizenzierten Programme in vielen Staaten, vornehmlich in Lateinamerika und Zentral-/Osteuropa, zum Alltag gehört. Hingegen sind die finanziellen Verluste in den Industrienationen im Vergleich zu Entwicklungs- und Schwellenländern enorm.

Zusammenfassung

Die Entwicklung neuer Kopierschutzverfahren, die lange genug Angriffen standhalten, schreitet immer weiter voran. Doch zeigt sich, durch verschiedenste Beispiele und Studien belegt, dass allein technische Versuche, sowie die Androhung drakonischer Strafen zur Eindämmung von Softwarepiraterie nicht ausreichen. Vielmehr sind Konzepte gefragt, die den Menschen, seine Wahrnehmung von Recht und Unrecht als auch seine Handlungen mit in die Problemlösung einbeziehen. So kann allein der Versuch, mit Nutzern illegaler Software oder mit Hackern und Crackern einen Dialog aufzubauen, dazu führen, Verständnis für die Situation von Unternehmen zu erlangen und evtl. ihr Handeln in legale Bahnen zu lenken. Selbst wenn dies nicht der Fall sein sollte, so können allein die Bemühungen den Respekt voreinander vergrößern.

Dieser Respekt ist es, der die Grundlage für Vertrauen schafft. Vertrauen und die Überzeugung, dass durch die richtigen Anreize ein faires Umgehen miteinander möglich ist, wurde im Bereich der kommerziellen Verteilung virtueller Waren das PotatoSystem (PS) erschaffen. Intention ist die Annäherung zwischen Konsumenten und Anbietern, denen ein Instrument zur gegenseitigen Unterstützung zur Verfügung gestellt wurde. Das zugrunde liegende Provisionsmodell dient als Basis aller Transaktionen.

Das PotatoSystem wurde um ein Teilsystem erweitert, das die Liste der zu erwerbenden virtuellen Waren um den Medientyp Software verlängert. Dabei konnte festgestellt werden, dass aufgrund der Intention des PS das Teilsystem zur elektronischen Softwareverteilung unter anderen Gesichtspunkten als bisherige kommerzielle Lösungen zu konzipieren war. Der Vertrag des PS als Vermittlungssystem involviert jeden Anbieter in den Wertschöpfungsprozess stärker als es andere kommerzielle Unternehmen mit Schwerpunkt auf eCommerce-Lösungen tun. Es muss immer bedacht werden, dass die Zielgruppen vom Hobby-Entwickler bis zum mittleren Unternehmen reichen. Deren Ausrichtung ist nicht (nur) streng kommerziell auf Profit ausgerichtet. Jeder Anbieter muss bestimmte Voraussetzungen erfüllen und die Inanspruchnahme seiner verfügbaren Online-Ressourcen in Kauf nehmen.

Die realisierte ESD-Lösung kann noch vielen Teilen verbessert werden. Dies ist i.A. abhängig von den Wünschen und Vorstellungen der 4FO AG. Es existiert jetzt eine Basis, an der weitere Entwicklungen anknüpfen können.

9 Ausblick

Der Hauptvertrieb virtueller Güter des PotatoSystems (PS) wird weiterhin bei Musik liegen und Software unter der Rubrik „Sonstiges“ geführt werden. Es ist natürlich nicht vorhersehbar, in wie weit die Möglichkeit einer ESD von Softwareentwicklern angenommen werden wird. Das PS als Vertriebsweg zu nutzen hat für Anbieter den Vorteil einer Outsourcing-Lösung. Übernommen wird die Auslieferung des Programms als auch der Produktschlüssel.

Zusätzlich können der *Community*-Charakter und die dadurch entstehenden *Super-Distribution-Channels* den Bekanntheitsgrad steigern und zu mehr Käufen führen.

Zu wünschen wäre eine Diversifizierung des bestehenden Provisionsmodells auf horizontaler Ebene. Bisher ist die Verteilung rein vertikal verzweigend. Es wird davon ausgegangen, dass das Verlangen eines Konsumenten nach Provisionen stärker ist als die einfache Weitergabe eine Ware mittels Kopie. Das bedeutet, dass theoretisch jeder Käufer seine Freunde und Verwandten für evtl. gleiche Geschmäcker bezahlen lässt. Er würde nicht eine Kopie bspw. eines Musikstückes weitergeben, sondern seinen Verkaufslink. Obwohl keine wissenschaftlichen Studien vorliegen, ist dieser Gedanke doch sehr befremdend. Würde stattdessen die Weitergabe auch auf horizontaler Ebene erlaubt, ergäben sich weitere Vertriebskanäle. Ein Beispiel: Zwei Konsumenten, nennen wir sie Alice und Bob, kennen sich persönlich. Alice ist das PotatoSystem bekannt, Bob hingegen nicht. Alice macht Bob auf einen Musiktitel aufmerksam und schenkt ihm eine Kopie. Nun hat Bob aber kein Interesse, den Musiktitel zu kaufen, weil er schon eine Kopie besitzt. Da er aber auch nicht als Wiederverkäufer agieren darf, könnte er den Titel nur noch über Tauschbörsen weitergeben. Der Verbreitungsgrad auf diesem Wege ist enorm und würde den des PotatoSystem mit Leichtigkeit überflügeln. Würde das PS über die Tatsache hinwegsehen, dass Bob keine legale Kopie besitzt, ihm aber trotzdem erlauben, als neuer Wiederverkäufer zu agieren, hätte der Musiker oder das Label zwar einen Verlust erlitten. Da Bob aber seinerseits nun legal weiterverkaufen darf, besteht durch ihn nicht mehr unmittelbar die Gefahr, dass der Musiktitel über Tauschbörsen veröffentlicht wird. Evtl. betreibt Bob eine Internetseite, die oft besucht wird. Vielleicht besitzt er sogar ein kleines Label. Er könnte somit den Bekanntheitsgrad des Musiktitels, des Labels, des/der Musiker/s und vor allem des PotatoSystems erhöhen. Bob benötigt nur einen Weiterverkaufslink, den bspw. Alice ihm ausstellt. Sie erweitert ihre TAN um ein Literal, z.B. a. Das PotatoSystem würde an diesem Zusatz erkenne, dass es sich hierbei nicht um einen legalen Käufer handelt und könnte ihm weniger Provision oder andere Einschränkungen (als kleine Bestrafung) auferlegen. Wie

Ausblick

dies alles konzeptionell und realisiert werden könnte, sollte Thema einer weiteren Diplomarbeit sein.

Eine Standardisierung des PS durch Anlehnung an bestehende Normen sichert gleich bleibende Qualität des Systems. Die Schaffung eines Quality-Management-Handbuchs (QMH) nach EN ISO 9000-3 würde einen vereinheitlichten Software-Entwicklungszyklus beschreiben, nach dessen Vorgaben sich alle Mitarbeiter und an dem System arbeitenden Entwickler zu richten hätten. Dies gilt sowohl bei der Einführung neuer Teilsysteme und Erweiterungen, als auch bei der Wartung und Pflege des Systems selbst.

Anhang A

Das hier vorgestellte Softwareprogramm Shareguard Locksmith ist ein Werkzeug, mit dessen Hilfe sich aus bereits bestehenden ausführbaren Programmen eine Sharewareversion erzeugen lässt. Entwickelt wurde es von der Firma Zapper Software (49). Dieses Produkt wurde ausgewählt, weil es den folgenden Anforderungen:

- online erhältlich
- einfach zu bedienen
- kostengünstig
- Schutz vor gewöhnlichen Hackerangriffen

genügt. Zu seinem Funktionsumfang gehört zusätzlich die Generierung einer Produktschlüsselliste als Textdatei und die das Erzeugen einer Installationsdatei. Einzige Einschränkung ist auch hier die Betriebssystem-unabhängigkeit, da es nur für Microsoft Windows generiert. Im Folgenden wird kurz das Erstellen einer Shareware gezeigt, ohne auf Details einzugehen. Diese können über die Hilfeseite des Produkts nachgelesen werden. Viele Voreinstellungen können übernommen werden.

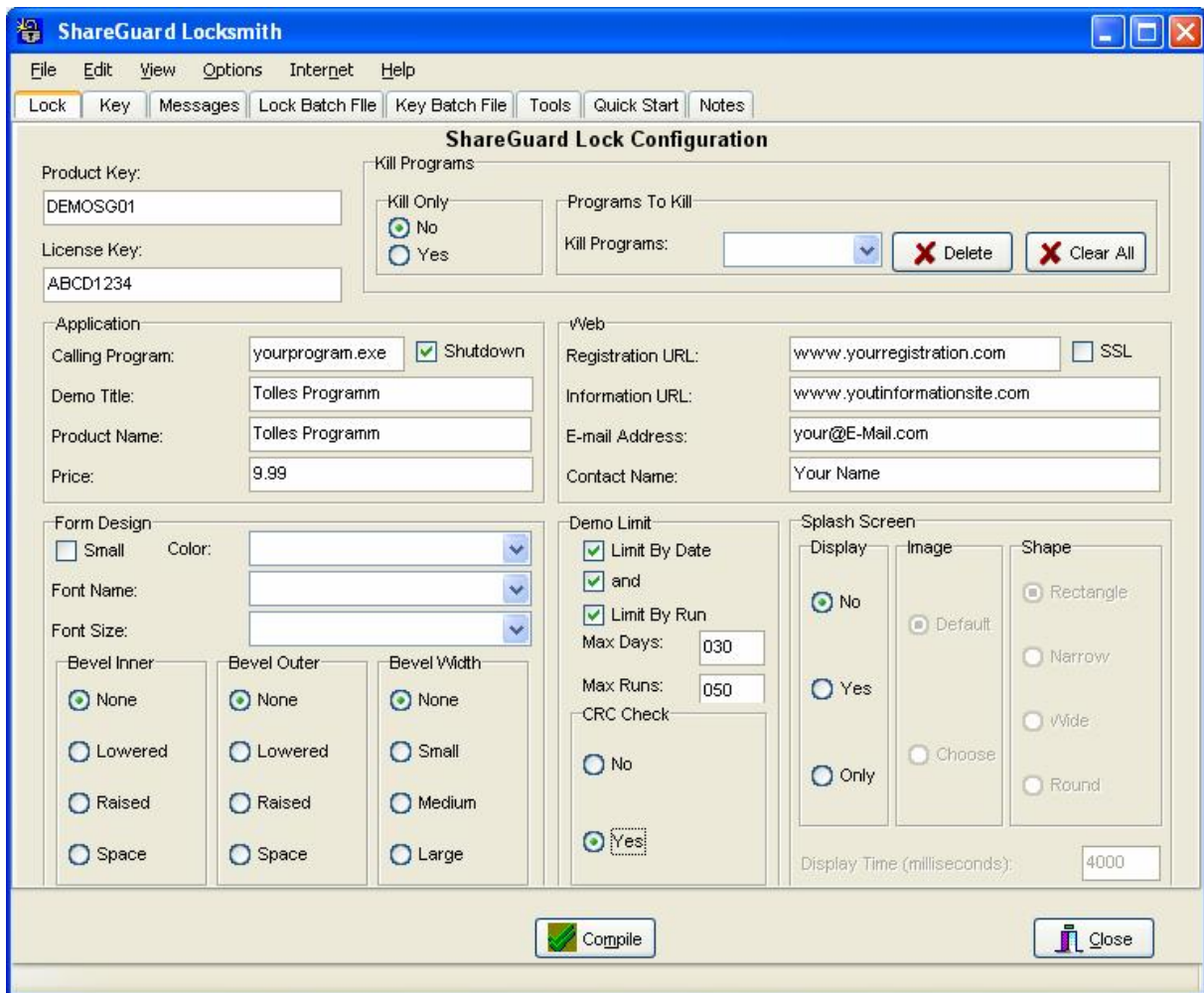
Zu Beginn wird die Eingabe eines für das Produkt *Product-Key* und eines *License-Key* gefordert. Beide Werte sollen für jedes Produkt eindeutig sein und haben eine eins-zu-eins-Beziehung. Screenshot 1 zeigt ein Beispiel. Anschließend werden die ausführbare Datei und alle zusätzlich benötigten Dateien ausgewählt. Die Auswahl wird mittels eines Dialogfensters vorgenommen, welches hier nicht abgebildet ist.

Ausgefüllt werden müssen noch die Namensfelder und der Preis. Die Cent-Beträge werden durch einen Punkt getrennt. Da es sich um ein Produkt einer kleinen Firma aus Kanada handelt, wird der Preis beim Anbieter mit vorangestelltem Dollarzeichen dargestellt. Potentielle Käufer sollten auf diesen Missstand hingewiesen werden.

Wichtig ist das Feld *Registration URL*. Hier wird die Anbieter-Internetseite eingetragen, die den vom PS generierten Verkaufslink enthält. Oder der Anbieter trägt nach der Registrierung den Verkaufslink ein und erstellt die Shareware nochmals (siehe Kapitel 6.3).

Es gilt nun über die Test-Perioden für Konsumenten zu entscheiden. Es gibt die Möglichkeit das Produkt über einen speziellen Zeitraum, eine bestimmte Anzahl an Starts oder beide Möglichkeiten in Kombination testen zu lassen. Entscheidet sich ein Anbieter für letzteres, muss bedacht werden, dass auch nach Ablauf der Zeitbegrenzung das Produkt ausführbar ist, falls noch Starts möglich sind.

Anhang A



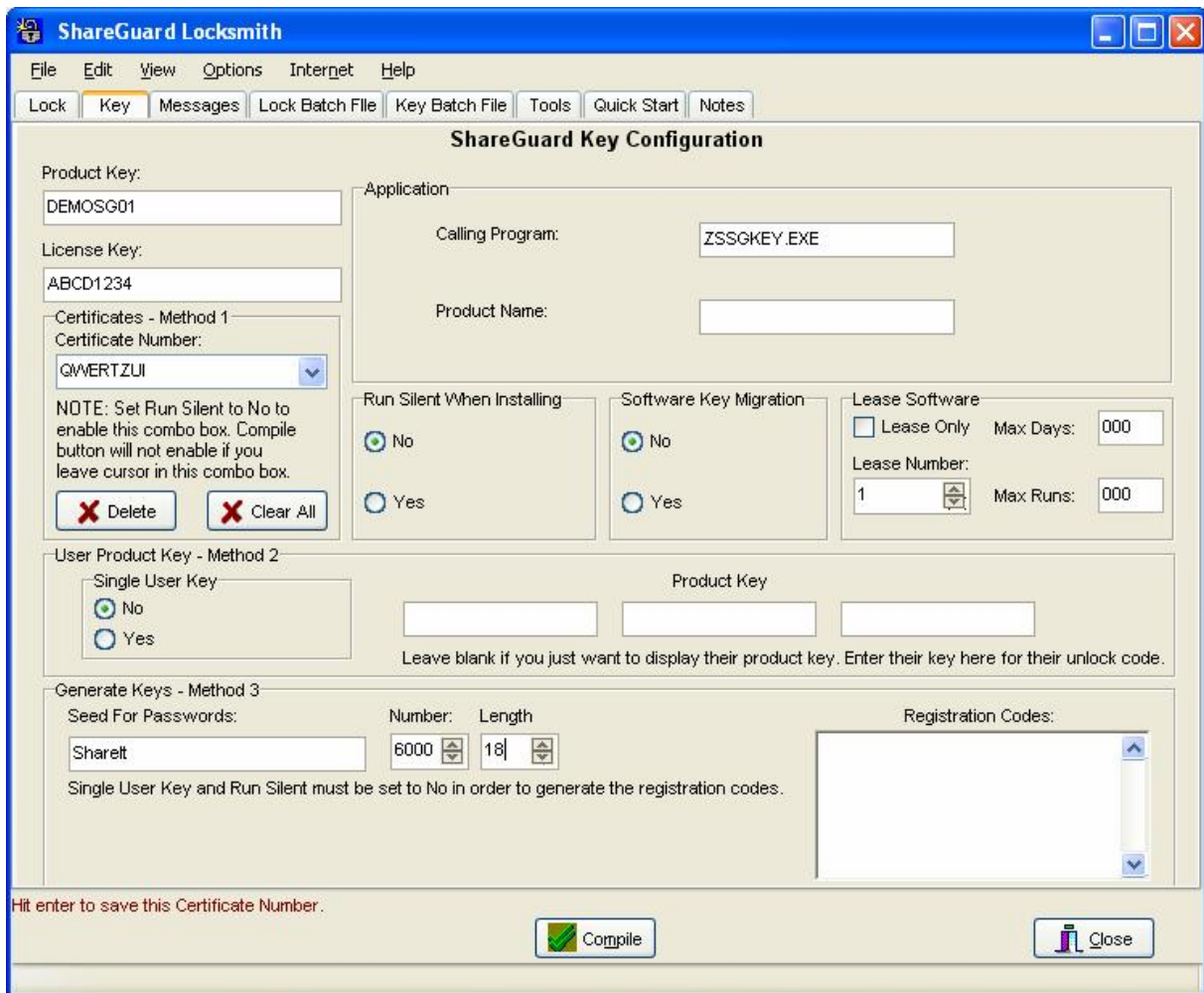
Screenshot 1 Shareware-Obfuscator - ShareGuard Locksmith - Reistrierkarte Lock

Die Aktivierung des CRC-Checks bietet einen Integritätstest und somit einen zusätzlichen Schutz vor möglichen Attacken durch Hacker.

Die folgenden Einstellungen sind für die Generierung der Produktschlüssel zur Auslieferung wichtig (ShareGuard bezeichnet sie als Passwort). Der folgende Screenshot 2 zeigt ein Beispiel. Auf der Seite der Registrierkarte *Key* werden der Eingaben der *Lock*-Registrierkarte übernommen. Eingegeben werden muss ein *Certificate*-Wert, der auf den Lizenz-Wert abgebildet wird.

Abschließend muss für die Generierung einer Schlüsselliste ein so genanntes *Seed* angegeben werden. Die wichtigste Angabe ist die Anzahl der zu erstellenden Schlüssel. Es kann nicht vorhergesagt werden, wie viele Konsumenten ein Produkt erwerben werden. Trotzdem sollte hier ein hoher Wert stehen, um im Notfall genügend Schlüssel ausliefern zu können. Weitere Informationen darüber sind in der Hilfedatei des Programms nachzulesen.

Anhang A



Screenshot 2 Shareware-Obfuscator - ShareGuard Locksmith - Registrierkarte Key

Sind alle Entscheidungen getroffen und alle Einstellungen vorhanden kann der Prozess über den *Compile*-Button gestartet werden. Dabei werden immer wieder Meldungen mittels Pop-up-Fenster angezeigt, die Benutzerinteraktion erwünschen. So kann man bspw. die generierte Shareware mit einem integrieren Mechanismus testen. Interessant ist zum einen die Möglichkeit eine Setup-Datei generieren zu lassen. Diese Option sollte angenommen werden, da alle benötigten Dateien in eine .exe-Datei generiert werden. Die folgende Frage, ob Käufer eine Online-Registrierung ermöglicht werden soll, muss beim Vertrieb über das PS abgelehnt werden, da sonst keine Schlüsselliste generiert würde.

Das nächste Fenster stellt den Anbieter vor die Entscheidung, die Registrierungsmechanismen in die kompilierte .exe zu integrieren oder als separates Programm zu erstellen. Letzteres hat den Vorteil einer flexibleren Registrierung und die Möglichkeit, weitere Produktschlüssel generieren zu können, die das ausgelieferte Programm freischalten. Der Nachteil dabei ist, dass zwei separate Programme existieren. In der ersten Variante existiert nur eine Datei, jedoch können keine weiteren Schlüssel zu Freischaltung generiert werden. Diese Entscheidung muss jeder Anbieter für sich selbst treffen. Zu bedenken ist dabei, dass das Produkt über das

Anhang A

PS nur als Ganzes ausgeliefert wird. Bei der zweiten Variante sollten beide Programmteile in ein Archiv gepackt werden, z.B. mittels WinZip, WinRar, etc. Unabhängig von der getroffenen Entscheidung kann nun mit der Generierung der Setup-Datei des eigentlichen Programms angestoßen werden und anschließend des Registrierungsmechanismus.

Im Installations-Verzeichnis von ShareGuard Locksmith findet sich nach der Herstellung ein Verzeichnis namens Output. Hier findet man eine Datei namens *Password.txt*, in der alle generierten Produktschlüssel zu finden sind. In zwei weiteren Unterverzeichnissen sind bei entsprechender Auswahl die Setup-Dateien für das Programm (Output) und des Registrierungsmechanismus (Key) enthalten. Sollte alles integriert worden sein, muss nur die Sharewaredatei im Unterverzeichnis „Output“ zum Erwerb angeboten werden. Alles weitere funktioniert wie in den Kapiteln 5.3.1, 5.5, 6.2.2 und 6.3 beschrieben.

Literaturverzeichnis

Literaturverzeichnis

- (1) 4FO AG; PotatoSystem – Das PotatoSystem; Homepage, <http://www.potatosystem.com/info/de/>, letzter Zugriff
- (2) 4FO AG; 4FriendsOnly AG – Was können wir alles bieten; Homepage, <http://www.4fo.de/de/>, letzter Zugriff
- (3) Holger Krauß; Konzeption und Realisierung der Server-Komponente für ein P2P-File-Sharing-System, bei dem die User am Umsatz beteiligt sind; Diplomarbeit, TU-Ilmenau, November 2002, http://www.4fo.de/download/PotatoServerDiplom_highres.pdf
- (4) Jens Hasselbach; Konzeption und Realisierung der Client-Komponenten für ein P2P-File-Sharing-System mit Umsatzbeteiligung für die Benutzer; Diplomarbeit, TU-Ilmenau, Mai 2002, http://www.4fo.de/download/Diplomarbeit_JHasselbach.pdf
- (5) Gabriele Frings; Konzeption und Realisierung eines plattformübergreifenden Web-Services zur Verwaltung von Nutzerkonten für virtuelle Waren; Diplomarbeit, TU-Ilmenau, Juni 2003, http://www.4fo.de/download/Diplomarbeit_Potato_Webservice.pdf
- (6) Dirk Behrendt; Konzeption und Realisierung einer Nutzerschnittstelle unter Einbeziehung von Web-Services für die Verwaltung von Nutzerkonten; Diplomarbeit, TU-Ilmenau, November 2003, http://www.4fo.de/download/diplomBehrendt_komprimiert.pdf
- (7) Oliver Lorenz; Konzeption und Realisierung eines anbieterunabhängigen Web-Services zur Autorisierung von Online-Zahlungstransaktionen; Diplomarbeit, TU-Ilmenau, März 2004, http://www.4fo.de/download/Diplom_Lorenz.pdf
- (8) Michael Kunze; Konzeption und Realisierung eines fairen Software-Kopierschutzes basierend auf einer Client/Server-Architektur; Diplomarbeit, TU-Ilmenau, September 2004, http://www.4fo.de/download/diplomarbeit_kunze.pdf
- (9) Jeannine Emer; Die Netzökonomie digitaler Güter am Beispiel des Potato-Systems -Entwicklung eines Informationssystems; Diplomarbeit, TU-Ilmenau; http://www.4fo.de/download/emer_diplomarbeit.pdf
- (10) André Hartmann; Konzeption und prototypische Realisierung einer Client-Komponente für die digitale Musikdistribution mittels personalisierter Inhaltsproben; Diplomarbeit, TU-Ilmenau, Oktober 2004, http://www.4fo.de/download/diplom_hartmann.pdf
- (11) Jürgen Nützel; Wie kann man mit dem Potato-System eine Ware verkaufen, die alle schon haben?; 2nd Media Workshop in Thuringia: "Digital Media Rights", May 20 and May 21 2003 in Erfurt, http://www.4fo.de/download/fktg_Nuetzel_final.pdf
- (12) Jürgen Nützel; Das PotatoSystem – mehr als Content Management und Bezahlservice für digitale Musik; Leipziger Informatik-Tage 2004, Leipzig, 29. September - 1. Oktober, 2004, in Von e-Learning bis e-Payment, Editor: K.-P. Fähnrich, K.P. Jantke, W.S. Wittig, ISBN: 3-89838-057-2, S. 113-122, http://www.4fo.de/download/LIT-2004-Nuetzel_090804.pdf
- (13) Jürgen Nützel, Rüdiger Grimm; Musikvertrieb mit Potato Web Services - Kaufanreize für Musik über die Web Services des PotatoSystems; DuD 3/2005, S. 125-129, Vieweg-Verlag, http://www.4fo.de/download/PotatoDuD0305_preprint.pdf
- (14) Rüdiger Grimm, Jürgen Nützel; Geschäftsmodelle für virtuelle Waren, DuD 5/2002, S. 261-266, Vieweg-Verlag, http://www.4fo.de/download/virwar_dud0502_preprint.pdf
- (15) Kathleen Biedermann; Das Geschäftsmodell des Potato-Systems –Analyse zur Entwicklung eines Geschäftsmodells; Projektarbeit, Institut für theoretische und technische Informatik, Fakultät für Informatik und Automatisierung, TU-Ilmenau; August 2003

Literaturverzeichnis

- (16) Jürgen Nützel; Matching Algorithms in alternative File-sharing Systems to find new Users having new Content; Innovative Internet Community Systems in Leipzig June 19-21, 2003, LNCS 2877, ISBN 3-540-20436-9, p. 180 – 188, http://www.4fo.de/download/PotatoMatching_Final.pdf
- (17) Patrick Aichroth, Jens Hasselbach; Incentive Management for Virtual Goods: About Copyright and Creative Production in their Digital Domain; International Workshop for Technology, Economy, Social and Legal Aspects of Virtual Goods, May 22 - 24 2003 in Ilmenau, Germany, http://virtualgoods.tu-ilmenau.de/2003/incentive_management.pdf
- (18) Tom Sheldon; Encyclopedia of Networking & Telecommunications; McGraw-Hill/Osborne Media, Mai 2002, ISBN: 0072120053
- (19) Eldad Eilam; Reversing: Secrets of Reverse Engineering; Wiley Publishing, Inc., 2005, ISBN-10: 0-7645-7481-7, ISBN-13: 978-0-7645-7481-8
- (20) Christian Collberg, Clark Thomborson, Douglas Low; A Taxonomy of Obfuscating Transformations; Technical Report #148, Department of Computer Science, University of Auckland; 1997; <http://www.cs.arizona.edu/~collberg/Research/Publications/CollbergThomborsonLow97a/index.html>
- (21) Christian Collberg, Clark Thomborson, Douglas Low; Manufacturing Cheap, Resilient, and Stealthy Opaque Constructs; Department of Computer Science, University of Auckland; 1998
- (22) Christian Collberg, Clark Thomborson; Software Watermarking: Models and dynamic Embeddings; Department of Computer Science, University of Auckland, 1998
- (23) Shareware Autoren Vereinigung SAVE; Homepage, <http://www.s-a-ve.com/>, letzter Zugriff: 31.05.2006
- (24) Stefan Kooths, Markus Langenfurth, Nadine Kalwey; Die Bedeutung der Microsoft Deutschland GmbH für den deutschen IT-Sektor (Economic Impact Study); Muenster Institute for Computational Economics (MICE), University of Muenster, 2003
- (25) Fundamental Modeling Concepts (FMC); Homepage, <http://www.f-m-c.org/>, letzter Zugriff: 04.06.2006
- (26) Bundesverband Informationswirtschaft Telekommunikation und neue Medien e.V. (BITKOM); Boom bei DSL-Zugängen in Deutschland; 07 November 2005; http://www.bitkom.org/de/presse/30739_34501.aspx
- (27) BITKOM– Arbeitskreis IT-Outsourcing Projektgruppe BPO; Business Process Outsourcing – Leitfaden – BPO als Chance für den Standort Deutschland; Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V.; 20. September 2005; http://www.bitkom.org/files/documents/BITKOM_Leitfaden_BPO_Stand_20.09.05.pdf
- (28) Jim Knopf; The Origin of Shareware; The Freeware Hallo of Fame, Rey Berry; Homepage, <http://www.freewarehof.org/sstory.html>; letzter Zugriff: 31.03.2006
- (29) Luke Hohmann; Beyond Software Architecture: Creating and Sustaining Winning Solutions; Addison Wesley, 30. Januar 2003; ISBN: 0-201-77594-8
- (30) Andrew M. St. Laurent; Open Source and Free Software Licensing; O'Reilly, August 2004; ISBN: 0-596-00581-4
- (31) Chris DiBona, Sam Ockman, Mark Stone; Open Sources: Voices from the Open Source Revolution; O'Reilly, 1999; ISBN: 1-565-92582-3
- (32) Volker Grassmuck; Freie Software zwischen Privat- und Gemeineigentum; Bundeszentrale für politische Bildung (bpb); Bonn; 2002
- (33) Business Software Alliance (BSA), International Data Corporation (IDC); Second Annual BSA and IDC Global Software Piracy Study; Mai 2005; <http://www.bsa.org/globalstudy/upload/2005-2006%20Global%20Piracy%20Study.pdf>

Literaturverzeichnis

- (34) Business Software Alliance (BSA), International Data Corporation (IDC); Expanding the frontiers of our digital future – Reducing software piracy to accelerate global IT benefits; BSA, IDC; Dezember 2005; <http://www.bsa.org/germany/piraterie/upload/IDC-Impact-Study-complete-english.pdf>
- (35) Wenbo Mao; Modern Cryptography: Theorie and Practice; Prentice Hall PRT, 25 Juli 2003; ISBN: 0-13-066943-1
- (36) Rikki Kirzner; Electronic Software Distribution: A Long Shot; UniForm Association – People advocating Open Computing, März 1996; Homepage: <http://www.uniform.org/publications/ufm/mar96/swdist.html>; letzter Zugriff: 02.04.2006
- (37) Trusted Computing Group; Homepage: <http://www.trustedcomputinggroup.org>; letzter Zugriff: 06.05.2006
- (38) Trusted Computing Group; Trusted Computing Group Backrounder; Trusted Computing Group, Januar 2005
- (39) Trusted Computing Group; TCG Specification Architecture Overview; Specification Revision 1.2; Trusted Computing Group, April 2004
- (40) Trusted Computing Group; Trusted Computing Group Work Group Charter Summary; Trusted Computing Group, 2005
- (41) Stefan Kreml: 22C3: Trusted Computing auf unsicherer Basis; heise online, Dezember 2005; Homepage: <http://www.heise.de/newsticker/meldungen/67809>
- (42) Bundesamt für Sicherheit in der Informationstechnik; Homepage: http://www.bsi.de/sichere_plattformen/trustcomp/index.htm; letzter Zugriff: 05.05.2006
- (43) Kristin Anderson, Carol Kerr; Customer Relationship Management; McGraw-Hill, 2002; ISBN: 0-07-139412-5
- (44) Y. Shafranovich; Common Format and MIME Type for Comma-Separated Values (CSV) Files; Network Working, Request for Comments: 4180, Category: Informational; Internet Engineering Task Force, Oktober 2005, <http://www.ietf.org/rfc/rfc4180.txt>
- (45) Mike Hartmann; Windows-Produkt-Aktivierung ausgehebelt; tecchannel, IDG Business Verlag GmbH; 16.07.2001; <http://www.tecchannel.de/client/windows/401698/index.html>
- (46) Alan O. Freier, Philip Karlton, Paul Kocher; The SSL Protocol Version 3.0; Transport Layer Security Networking Group, Internet Draft; Netscape Communications, 18. November 1996; <http://wp.netscape.com/eng/ssl3/draft302.txt>
- (47) T. Dierks, C. Allen; RFC 2246: The TLS Protocol, Version 1.0; Network Working Group, Category: Standards Track, Januar 1999; <http://www.faqs.org/rfcs/rfc2246.html>
- (48) J. Callas, L. Donnerhacke, H. Finney, R. Thayer; OpenPGP Message Format; Network Working Group, Request for Comments: 2440, Category: Standards Track; Internet Engineering Task Force, November 1998; <http://www.ietf.org/rfc/rfc2440.txt>
- (49) Zapper Software – Association of Shareware Professionals Development Member; Homepage: <http://www.zappersoftware.com/>; letzter Zugriff: 07.06.2006
- (50) Bruce Johnson; Software Copy Protection; CapeSoft, 29. Juni 2001; <http://www.capesoft.com/opinion/CopyProtection.htm>
- (51) element5 – a digital river company, Unternehmensprofil; Homepage: <http://www.element5.com/aboutus.html>; letzter Zugriff: 02.06.2006
- (52) PACE Anti-piracy; Homepage: <http://www.paceap.com/>; letzter Zugriff: 04.06.2006

Literaturverzeichnis

- (53) Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides; Design Patterns – Elements of reusable object-oriented software components; Addison Wesley; Juli 1997; ISBN: 0201633612
- (54) Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts; Refactoring: Improving the design of existing code; Addison Wesley; Juni 1999; ISBN: 0201485672
- (55) Chris Anley; Advanced SQL Injection in SQL Server Applications; An NGSSoftware Insight Security Research (NISR) Publication; Next Generation Security Software Ltd; 2002;
http://www.nextgenss.com/papers/advanced_sql_injection.pdf
- (56) das ELKO - das ELEktronik-KOmpendium.de; DVD-Typen; Elektronik-Kompendium;
<http://www.elektronik-kompendium.de/sites/com/1001121.htm>
- (57) ComponentSource – The definitive source of software components; Homepage:
<http://www.componentsource.com/index.html>, letzter Zugriff: 07.06.2006
- (58) Industrial Business Mashines; ASCC Reference room; IBM; Homepage: http://www-03.ibm.com/ibm/history/exhibits/markI/markI_reference.html; letzter Zugriff: 09.06.2006

Eidesstattliche Erklärung

Eidesstattliche Erklärung

Hiermit erkläre ich, Martin Fürstenau, an Eides statt, dass ich diese Diplomarbeit selbstständig und nur unter Verwendung der im Literaturverzeichnis angegebenen Quellen angefertigt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Potsdam, den 13.06.2006

.....

(Martin Fürstenau)