Software Kolumne in 'Finanz und Wirtschaft'

Prof. J.A. Makowsky
Department of Computer Science
Technion - Israel Institute of Technology, Haifa, Israel
and
Scientific Consulting, Zurich, Switzerland

April 23, 2009

Contents

Chapter 1

Computeranimation:

HighTech-Trickfilm oder Aufbruch in neue Visualisierungsmöglichkeiten ?

(Oktober 1988)

Computergraphik findet mehr und mehr Eingang in die verschiedensten Anwendungen digitaler Technologien. Computererzeugter dreidimensionaler Trickfilm ist dabei nur die spektakulärste Manifestation. Ihr perfekter Realismus steht dabei im Widerspruch zu anderen Anwendungen von computererzeugten Animationen, bei denen Visualisierungsprobleme in der Darstellung sehr komplexer Daten eine zentrale Rolle spielen. Langfristig werden Soft warepakete wohl interaktive Animationen sein, die man auf einem Computer abspielt wie eine Schallplatte auf einem Grammaphon.

1.1 Digitalisierung der Bildfläche

Die erste Idee ist einfach: Bilder, die gedruckt oder projiziert werden können, bestehen aus Einzelpunkten - Pixel - mit bestimmten Farbwerten und "verschwimmen" erst durch das begrenzte Auflösungsvermögen des Betrachters zu kontinuierlichen Bildern. Dasselbe gilt auch für bewegte Bilder, die pro Sekunde meistens aus 25 festen Bildern bestehen. Dass die Menge der Bildpunkte dabei riesig wird, macht das Erzeugen von solchen Bildern "von Hand" fast unmöglich, ist aber mit Hilfe von Computern mit immer grösserer Datenverarbeitungskapazität nicht nur möglich, sondern scheint eines der idealen Anwendungsgebiete für leistungsstarke Computer zu sein. Und tatsächlich ist zum Beispiel die alljährlich stattfindende Computergraphik-Konferenz SIGGRAPH in den USA die wohl grösste Veranstaltung dieser Art in der Computerbranche, wo dieses Jahr ca. 20'000 Teilnehmer zu einer Mischung von Messe, Filmfestival und wissenschaftlicher Konferenz drängten. Computergraphik, d.h. computererzeugte Bildherstellung, ist publikumswirksam und zugleich testtauglich für die Soft- und H ardwarebranche und spielt somit eine ähnliche Rolle für die Computerindustrie wie Autorennen für die Autoindustrie der 50er Jahre. Die Anwendungen reichen vom comicstrip-ähnlichen Computerspiel bis zur Herstellung von Kurzfilmen in Kinoqualität, vom Desktop Publishing auf PC-Niveau bis zu CAD/CAM-Systemen der Grossindustrie, wo die Verwendung von Supercomputern nicht abwegig sondern notwendig ist.

Trotzdem ist Computergraphik guter Druck- oder Projektionsqualität sehr teuer und viele der neulich gegründeten Gross- und Kleinunternehmen verschwinden nach kurzer Zeit wieder vom Markt, weil ihre Markterwartungen sich als falsch erwiesen

oder weil sie ihre eigenen Produktionskosten falsch einschätzten. Es fällt auch auf, dass der markbewussteste Riese der Computerbranche, IBM, auf diesem Gebiet eine eher bescheidene Rolle spielt und sich bis jetzt nicht durch öffentliche Anstrengungen, auch diesen Markt zu erobern, bemerkbar gemacht hat. Diejenigen Unternehmen, die sich auf dem Markt halten können, haben meist ein zusätzliches Standbein, sei es in der Filmindustrie oder Hardwareproduktion. Kleinere Computerfilunternehmen ohne eigene Forschungs- und Entwicklungsabteilung riskieren durch den rasendschnellen technischen Fortschritt leicht aus dem Wettbewerb geworfen zu werden.

1.2 3-D-Graphik und 3-D-Animation

Bilder, die wir betrachten sind zweidimensional. Trotzdem spricht man von dreidimensionaler (3-D) Computergraphik und -animation. Diese Sprechweise bezieht sich nicht auf das dargestellte Bild, sondern auf die Darstellung der Daten, die zur Erzeugung des dargestellten Bildes führen. In der 3-D-Graphik werden die Objekte als Mengen von Koordinatenpunkten in einem 3-dimensionalen Vektorraum gespeichert, wobei alle Methoden der darstellenden Geometrie angewendet werden können. Darüberhinaus können die Objekte gefärbt werden und die Oberflächenpunkte mit Lichteigenschaften (Brechung, Spiegelung, Strahlung von Licht) versehen werden. Schliesslich wird ein zweidimesionales Bild errechnet als Projektion auf eine durch die Position des virtuellen Betrachters gegebene Bildfläche, wobei, je nach Rechenaufwand, verschiedene Beleuchtungsmodelle auch Lichteffekte miteinbeziehen. Die aufwendigste Methode der Bildberechnung verwendet das Raytracing, wobei alle Lichtstrahlen im Modell verfolgt und ihr Auftreffen auf die Bildfläche ausgerechnet wird. Bei einer Auflösung von 2000 mal 2000 Bildpunkten (Spielfilmqualität) kann das Berechnen eines Bildes bei grosser Anzahl von Objekten oder entsprechend komplizierter Ausformung auch auf einem Grosscomputer mit extremer Berechengeschwindigkeit mehrere Stunden echte Rechenzeit betragen. Falls die Anwendung dies nicht erlaubt (Kosten oder Zeit) muss entweder Spezialhardware verwendet werden (Flugsimulatoren) oder es müssen Vereinfachungen im Beleuchtungs- und/oder Darstellungsmodell vorgenommen werden. Geschickt gewählte Vereinfachung, die mit dem Verwendungszweck des dargestellten Bildes in Einklang stehen, können dabei entscheidende Kosten- und Zeitersparnisse ergeben. Letztere sind wesentlich, wenn es sich um 3-D-Animation handelt, wo ja 25 Bilder pro Filmsekunde berechnet werden müssen.

1.3 Messen und Befragen

Die Verwendung so komplexer Daten zur Bilderzeugung nur schöner, effektvoller Bilder mag zwar reizvoll sein, scheint aber dieser Technologie langfristig nicht gerecht zu werden. Erst die interaktive Verwendung der Bilder macht diese Technologie interessant. Dazu gehört aber nicht nur die Animation (Bewegen der Objekte und des Blickwinkels, Veränderung der Beleuchtung) sondern auch das Befragen der Situation. Einfachste Fragen wären zuerst einmal Messdaten der Geometrie (Länge eines Objekts, Durchmesser, Distanz von einem anderen Obkekt), aber auch anderer physikalischer Zusammenhänge, die in die Dartsellung der Daten eingegangen sind (Gewicht, Elastizität, Reibung, optische Effekte), gegebenenfalls auch chemische oder ökonomische Eigenschaften, weiters auch Lösungen zu Optimierungsaufgaben in Planung, Konfiguration, Alternativen, kurz alles was aus den eingegebenen Daten berechenbar ist. Ob der Benützer die Rechnung vorformulieren (programmieren) oder nur die Eigenschaften der Lösung beschreiben (spez-

ifieren) muss, hängt von der Abfragesprache ab, die dem System unterlegt ist, und der Gesamtheit der Abfragen, die im System vorkonzipiert wurde. Die Antworten können dann auf dem Bildschirm auch, aber nicht nur, graphisch dargestellt werden.

Abfragesprachen wurden vor allem im Zusammenhang mit Datenbanksystemen und entwickelt, wo grosse Datenmengen in relativ einfacher Form von Relationen (Tabellen) dargestellt werden. Aber das Befragen des Systems, sei es nun Datenbank, Expertensystem oder Designsystem für Software oder industrielle Produktionsgüter, gehört zu den grundlegenden Manipulationsmöglichkeiten von Daten. Es ist deshalb umso erstaunlicher, dass das Befragen nicht allgemein zu den Benützermöglichkeiten beliebiger Softwarepakete gezählt wird. Fragen können alles betreffen, und beim Entwurf von Softwarepaketen muss natürlich auch der Bereich der möglichen Abfragen mitkonzipiert werden. Antworten zu Fragen sind aber nicht nur Ja/Nein, sondern oft Mengen von Datenkonfigurationen wie "alle Objekte, die....". Im Falle von Datenbanken sind die Antworten neue Relationen oder statistische Aussagen, aber im Falle von komplexeren Datenmengen müssen die Antworten auf geeignete Weise graphisch dargestellt, eben Visualisiert werden.

1.4 Visualisierung von Prozessen

Im Grunde genommen ist jede interaktive Software auch Animation, wenn sie visualisiert wird. Im einfachsten Fall blinkt das Lämpchen, das anzeigt, dass das System arbeitet. Es können auch mehrere "Lämpchen" blinken, die dann anzeigen, in welchem Programmsegment der Computer sich gerade befindet oder wieviel Zeit und Speicherplatz das Programm gerade beansprucht. All das sind auch "Messdaten" eines laufenden Programms. Von Visualisierung spricht man aber ernsthaft erst, wenn es darum geht grosse Datenmengen darzustellen in einer Art, dass die globalen menschlichen Sehfähigkeiten zur Geltung kommen. Diese Sehfähigkeiten erlauben uns durch Verwendung von Farbe, Topologie und Anordnungen Sinnzusammenhänge kompakt darzustellen und auch sie schnell wieder zu erfassen. Dabei spielt eine grosse Rolle, dass solche Bilddartsellungen, seien es Symbole, Ikonen, Notationen, graphische Dartsellungen oder ganze gefärbte Flächen, vom Menschen sehr schnell durch das blosse Ansehen von Beispielen erlernt werden können. Es ist allerdings zu beachten, dass solche Darstellungen ebenso irreführend sein können wie erhellend, vorallem dann, wenn neue Begriffsinhalte in ihrer Darstellung mit Sehgewohnheiten kollidieren, die andere Inhalte suggerieren. Wir alle kennen das bei der Darstellung von funktionalen Zusammenhängen (z.B. Preisentwicklung in der Zeit), wo ungewöhnliche Koordinatenwahl sowohl klä rend (Zeit von links nach rechts, linear; Preis von unten nach oben steigend) als auch verwirrend (Zeit nicht linear, Preis von oben nach unten im reziproken Wert) dargestellt werden können.

Fortschritte in der Mathematik und den exakten Wissenschaften sind oft mit Durchbrüchen in der Notation und graphischen Darstellung verbunden. Algebraisierung von Gleichungssystemen, Darstellende Geometrie oder die verschiedenen Atommodelle und Darstellungen von Molekularstrukturen liefern da gute Beispiele. Und so erstaunt es nicht, dass das Aufarbeiten solcher Durchbrüche und deren Integration in die interaktive Software wesentlich dazu beitragen, Benützerschnittstellen zu verbessern. 3-D Animation ist da bei nur ein, wenn auch spektakuläres Beispiel. Es ist das allgemeine Visualisierungsproblem, das bei zunehmender Computerisierung der Produktion, Verwaltung und Dienstleistung im Vordergrund steht, und dessen Lösung letztlich über die Anwendbarkeit computergesteuerter Methoden entscheidet.

1.5 Zukunftschancen und Markt

Mit der Verbesserung der graphischen Kapazitäten des Computers wird der Computer zu einem neuen Medium, vergleichbar dem Plattenspieler oder Videogerät, auf dem statt Musik oder Film eben interaktive Software abgespielt wird. Der Benützer kann dabei das System befragen und auf es auch einwirken, sei es durch Eingabe von zusätzlichen Daten oder Anweisungen. Graphik vergrössert dabei die Schnittstelle zwischen Benützer und Maschine. Die Möglichkeiten sind nur durch die Phantasie der Ingenieure und die benötigten Rechenzeiten beschränkt. Die Akzeptanz der neuen Methoden aber hängt sehr stark vom Benützerkomfort ab, der durch geeignete Visualisierungen stark gefördert werden kann. Computererzeugter Film ist hier nur eine Anwendung, dessen Kosten oft mehr durch den Tech-Appeal als durch die effektive Leistung gerechtfertigt werden muss. Der langfristige Markt für den Vertrieb und Verkauf von computererzeugtem Film liegt in seiner interaktiven Ausgestaltung als Expertensystem und im computergestützten Unterricht. Dabei ist wesentlich, dass wir lernen neue interaktive Geschichten zu erfinden und zu visualisieren ohne sie durch barock-überflüssigen Bilderreichtum im Keim zu ersticken. Die Forschungsergebnisse in Cognitiv Science, Künstlicher Intelligenz, Datenstrukturen und Algorithmik werden dabei eine wesentliche Rolle spielen und über die Verwirklichungsmöglichkeiten verschiedenster Traumprojekte ihr Urteil sprechen.

Chapter 2

Viren im Computer - ein soziales Symptom

(31.12.88)

Viren in Computern können Schaden anrichten. In leichteren Fällen beläuft sich der Schaden auf einige Stunden Arbeitsausfall, Reformatieren der Festplatte und viel Aerger im Bauch. in schlimmeren Fällen ist Schaden denkbar, dessen Ausmass schwer abzuschätzen ist und der in Geld umgerechnet Milliardenbeträge auch übersteigen kann. Zu Recht sind Verantwortliche von Rechenzentren und computerintensiven Betrieben verunsichert. Sie bringen die Virusgefahr in Zusammenhang mit den neuen, noch nicht in ihrer vollen Tragweite erfassten Technologien, die im weitesten Sinne mit Software zu tun haben.

Nun stimmt es, dass das Phänomen Software von unserer Kultur noch nicht in allen seinen Aspekten verstanden wird. Software ist manchmal ein Produkt, manchmal eine Dienstleistung und manchmal ein Stück Unterhaltungsliteratur, Fiction würde man im Englischen sagen. Software ist manchmal ein Stück Hardware, die Uebergänge sind fliessend. Aber das waren sie schon früher. Was ist ein Kochbuch ? Ein Kilo Papier, das brennt und als Wurfgeschoss tödliche Folgen haben kann; ein Nachschlagewerk mit Gebrauchsanweisungen, eine schriftliche fixierte Dienstleistung also; schliesslich auch ein Stück Literatur, mit Geschichten und Geschichte.

Computerviren sind auch Software. Aber sie sind primär *Sabotagemittel*, die in diesem Falle aus Software bestehen. Ein anderes sehr wirksames Sabotagemittel ist Zucker, ein Genussmittel des Alltags. Man stelle sich vor, jugendliche Banden (oder ausgeflippte Rentner oder Arbeitslose) würden sich einen Spass daraus machen, in die Benzintanks parkierter Autos Zucker zu schütten. Wir würden kaum den Gebrauch von Zucker oder von Autos, oder die chemischen Eigenschaften von Zucker und Benzin, in Frage stellen, sondern geeignete Schutzmassnahmen suchen, Schutzmassnahmen im technischen Bereich (verschliessbare Tankdeckel) und im sozialen Bereich (verschärfte Repression oder eine Erweiterung der Sozialpolitik). Für einzelne Fälle von Sabotage zeigt die Gesellschaft oft auch Verständnis: Vor Jahren hatte in Zürich, nach vierzig Jahren heimlicher Frustration, der Hauswart der Telefonzentrale Hottingen, ein Herr Hürrlimann, Feuer gelegt. Das ganze Telefonnetz der City war für vierzehn Tage lahmgelegt. Die Geschäftswelt reagierte mit berechtigtem Zorn, in den Medien erschienen Artikel, die sich über menschliche Kommunikation im Telefonzeitalter ausliessen und in der ehrlich arbeitenden Bevölkerung wurde für Hürrlima-Fürlima als Volksheld und Opfer der entfremdeten Arbeitsverhältnisse Sympathie und Mitleid empfunden. Die Justiz erklärte ihn für verrückt und man steckte ihn in eine geschlossene Anstalt.

Schutzmassnahmen für Computerviren gibt es viele, aber die meisten rühren an

den herrschenden Gewohnheiten, wie man mit Computern als Geräten umzugehen hat. Man muss Inputgeräte verschliessen wie Tankdeckel. *Was den sozialen Bereich betrifft, haben die Wenigsten verstanden.* Viren werden von Hackern in Umlauf geschickt, meist sehr junge oder mindestens infantile Leute, die viel Geschick und wenig Verständnis im Umgang mit Computern vorzuweisen haben. Hacker organisieren sich in Banden und verstehen sich als Vorkämpfer der Computerkultur. Sie verbrämen ihre Bandentätigkeit mit Ideologien und geben sich einen sozialen Auftrag. In vielen Aspekten sind sie die Halbstarken des Computerzeitalters. Und wie die Uebergänge zwischen Hell's Angels and Black Panthers fliessend sind und aus den Jugendrevolten auch echte politische Bewegungen hervorgegangen sind, so gibt es unter den verschiedenen Hackergruppen auch solche mit echtem politischen und gesellschaftskritischem Bewusstsein.

Die politisierte Hackerbewegung gibt vor, Schwachstellen unserer Gesellschaft im Umgang mit Computern aufzuzeigen. Aber vielmehr zeigt sie Schwachstellen auf in der beruflichen Organisation der Computerfachleute. Mediziner und Juristen schützen sichund die Gesellschaft mit gewissem Erfolg vor Missbrauch ihres Wissens und gegen Scharlatane mit straffen Berufsorganisationen und Berufsausübungslizenzen. Noch im letzten Jahrhundert kämpften verschiedene medizinische Schulen um das Monopol der Berufsausübung. Quacksalber nannte man immer die anderen, die, denen man Berufsehre absprechen wollte. Was die Hackerbewegung wirklich aufzeigt, ist, dass unter den Computerspezialisten die Unterscheidung in zünftige Fachleute und Quachsalber noch nicht abgeschlossen ist, dass noch keine Zunft oder Computerkammer existiert, die Zünftigkeit mit Sanktionen durchsetzen kann. Sabotage durch Fachwissen wird in der Gesellschaft durch Konsens und vertrauen vermieden. Das Vertrauen der Bevölkerung in die Stände der Fachleute wird seinerseits durch interne Sanktionen der privilegierten Stände gegen groben und offensichtlichen Missbrauch erworben und gesichert.

Ansätze zu solchen Berufsorganisationen gibt es schon, nationale und internationale. Die internationale, in den Vereinigten Staaten beheimatete Association for Computing Machinery, ACM, hat einen Ehrenkodex entworfen und bildet das seriöseste Forum für gerade diese Diskussion. Aber solange keine einheitliche Standesprüfung besteht, die vor dem Gesetz und in den Augen der Klienten als einzige anerkannt wird, ist die Gesellschaft vor Scharlatanen und randalierenden Hackern nicht zu schützen. Viren und andere Softwaresabotagemittel, die in Umlauf kommen, erfüllen eine wichtige soziale Funktion. Aber weder die politische motivierten Hacker noch die meinungsbildenden Halbexperten, die sich in der Presse zu diesen Themen äussern, erfüllen diese Funktion. Beide verwechseln die Pioniertatetigkeit, die Aneignung von Freiräumen im Stile des Wilden Westens und deren romantische Aura, mit der eigentlichen Problematik: dem Schaffen eines sozialen und rechtlichen Rahmens, indem diese neuen Räume verwaltet und genutzt werden können.

Chapter 3

Read, write, execute

(21.1.89)

Programme kann man schreiben, lesen und ausführen lassen. Der Computer liest sie in Binärcode, und führt sie aus. Er übersetzt sie automatisch von der Programmiersprache der Benützer, meist menschliche Geschöpfe, die in einem kulturellen Zusammenhang leben und kommunizieren, in seine Sprache. Er liest diese Programme nicht eigentlich, das heisst, er ordnet ihnen keine Bedeutung zu, sondern verlässt sich ganz auf die Syntax ihrer Sprache. Der Programmierer schreibt die Programme, so ist das üblich, nur für den Computer, und denkt dabei, wenn überhaupt, nur an den Benützer im Zusammenhang der Mensch-Maschine Beziehung, des Dialogs zwischen Benutzer und Programmschnittstellen. Dass die Computerkultur aber einen globaleren Zusammenhang konstituiert, in dem Computer und deren Programme gewartet werden müssen, in dem mündliche und schriftliche Traditionen etabliert werden, die länger leben werden als Programmierer und Programme, dass das kulturelle System Mensch-Computer seine eigenen Gesetzmässigkeiten hat und weiterhin haben wird, scheint im Verhalten der Träger der Computerkultur nicht nur keine Rolle zu spielen, sondern das Uebersehen dieses Zusammenhangs scheint mir eine der zentralsten Ursachen dafür zu sein, was wir, mangels besseren Verständnisses, als Softwarekrise bezeichnen und beklagen.

Man stelle sich eine menschliche Kultur vor, in der in den Schulen erst das Schreiben gelehrt und gelernt wird, und nicht das Lesen. Lesen hat eine sprachbildende Funktion: Wenn wir uns zum ersten Mal auf einem neuen Gebiet ausdrücken, haben wir immer Schwierigkeiten. Wir sprechen und schreiben erst einmal die Sprache anderer. Wir formulieren Sätze, die wir gehört und gelesen haben, wir modifizieren sie zaghaft, und nur wenige von uns formulieren wirklich vormals Unformuliertes. Wir denken in vorgeformten Gedanken und Geschichten, wir erleben die Welt um uns in vorgeformten Erfahrungsmustern. Das Zuhören, und später das Lesen, hat uns eben diesen Schatz an Erfahrungsmustern wesentlich erweitert. Dabei spielen Mythen, Lieder, Gedichte, Geschichten, Romane, aber auch Gebrauchsanweisungen, Gesetzestexte und Verordnungen, politische Berichterstattung und Buchhaltungsformen des Besitzes und der Gefühle gleichberechtigte Rollen. Sie alle bilden den Rahmen, in dem soziale und epistemische Strukturen gewartet und verändert werden, über Generationen hinaus in zeitlichen Dimesionen, die wir Geschichte nennen.

Wenn wir nun Kinder oder Erwachsene zum ersten Mal mit der Handhabe von Computern konfrontieren, ist es wesentlich, dass nicht schon bei dieser ersten Begegnung die Weichen falsch gestellt werden. Insbesondere sollten die computerunterstützte Tätigkeit des Texte- oder Programmeschreibens nicht mit dem Programmieren selbst verwechselt werden. Wer erfolgreich den Basiceditor bedienen und ein paar Zeilen Basic zum Laufen bringen kann, hat noch nicht pro-

grammiert, so wie einer, der auf dem Klavier interessante Geräusche produzieren kann, deswegen nicht unbedingt schon Musik macht. Musik entsteht erst, wenn sie auf tradierten oder tradierbaren Formen beruht, sei es als literarische, das heisst notierte und eben lesbare Musik, sei es als gespielte und nachspielbare Musik. Programmieren ist auch eine literarische Tätigkeit. Der Programmierunterricht muss zwar computerunterstützt erfolgen, aber auch mit einer Programmierfibel, einer Sammlung von einfachen und gut geschriebenen Programmen, die man lesen und abspielen kann, die man aber auch modifizieren und zusammensetzen kann. Zum Programmtext gehört auch die Dokumentation, die sich aus Benützerdokumentation, Wartungsdokumentation und einem Modulplan zusammensetzen sollte. Programme schreiben sollte eigentlich mit dem Schreiben von Dokumentationen beginnen und mit dem Schreiben von Code aufhören, nicht, wie in der Praxis üblich, umgekehrt. Software ist eine strukturierte Sammlung von Programmen mit einer transparenten Benützerschnittstelle, transparenter Dokumentation und transparent strukturierten Aufbau.

Erst wenn wir Software auch lesen und als Lesestücke zu schätzen gelernt haben, erst wenn wir Software nicht nur für Computer, sondern auch für menschliche Leser schreiben, wird die jetzige Phase der Alphabetisierung der Computerbenützer abgeschlossen sein. Denn so wie viele, die zwar Buchstabieren können, funktionelle Analphabeten sind, so sind leider auch viele Programmierer zwar Codeschreiber, aber weit davon entfernt, die Verhaltensweise der von ihnen verfassten Programme zu überblicken. Man ist ausserdem zu leicht dazu geneigt, die Schuld an der Unlesbarkeit von Programmen den Programmiersprachen in die Schuhe zu schieben. Es ist zwar richtig, dass Grammatik, Formenreichtum und Grundbegriffe allgemein, und der Programmiersprachen im Besonderen, auch die Ausdrucksweise der Sprachbenützer beeinflussen. Aber auch in Assembler lassen sich klare, und in Pascal konfuse Programme schreiben. Eine Schreib- und Lesekultur braucht Musse, um Blüten treiben zu können, Musse, Geduld, und langfristige Planung. Man muss erst zuhören und lesen lernen, bevor man schreiben darf, auch wenn man Software schreiben will.

Chapter 4

Die edlen Hacker und der Staatsanwalt

(7.3.89)

Der Computer Chaos Club Hamburg macht wieder und wieder internationale Schlagzeilen. Von der Titelseite der *New York Times* bis zur angesehensten Fachzeitschrift der Informatiker *Communications of the ACM* beschäftigen sich Journalisten und Fachleute mit der Analyse und den Folgen spektakulärer Einbrüche, die die Hacker aus Hamburg in den verschiedensten Computersystemen ausgeführt haben. Ich hatte neulich Gelegenheit, mich mit einigen Mitgliedern der Hamburger Hackerzunft ausführlichst zu unterhalten. Das widersprüchliche Selbstimage, das sie sich geben möchten, hat auf den ersten Blick durchaus attraktive Züge. Ueber die Gefahren, denen sie sich aussetzen, wurden sie sich erst durch die Aktivitäten der Staatsanwaltschaft bewusst. Jetzt droht ihnen ein Spionageprozess.

Die vorgebrachten integren Motive der Hacker, die Hackerethik, wie sie es nennen, stützen sich auf zwei Prinzipien: Information gehöre allen, Informationsmonopole seien a priori verwerflich und der Kampf gegen diese Informationsmonopole dürfe im Interesse des Volkes und getragen von dessen Angst im Umgang mit der Technolgie, auch mit unlauteren, aber von der Justiz noch nicht voll erfassten Mitteln geführt werden. Sie auferlegen sich auch ehrbaren Einschränkungen bei dieser Aktivität. Daten, so postulieren sie, werden grundsätzlich nur kopiert und nie verändert, Eingriffe in die Betriebssyteme sind zu vermeiden, überhaupt soll, nach der Auffassung des edlen Hackers, kein Schaden entstehen, sondern bestenfalls das Gemeinwohl gefördert werden.

Die Philosophie der edlen Hacker hat ihren Ursprung in der im Umfeld der 68er Bewegung entstandenen Bewegung "Computer for the People" der USA, die Computer- und Kommunikationsresourcen auch als Gemeingut betrachtete wie die Luft zum Atmen. Nun ist eine solche Erweiterung des Begriffes Gemeingut durchaus eine mögliche Utopie, im Islam sind theoretisch Zinsen verboten und für Wasser und Holz zum Feuern dürfen keine Gebühren erhoben werden. Aber unsere industrialisierte Gesellschaft erlaubt das exklusive Nutzen von Naturschätzen und Information und das Erheben von Gebühren für fast alles. Sicher werden wir bald auch eine Atemluftsteuer einführen, deren Höhe nach der Reinheit der Luft am Wohnort bemessen werden wird.

Aber auch wenn wir den edlen Hackern lautere Motive zusprechen wollen, so bleibt doch vieles problematisch. Insbesondere unterschätzen sie, aber auch oft die Hüter der bestehenden Gesetze, die Möglichkeiten der Justiz gegen das Tun und Treiben der Hacker im Rahmen der bestehenden Rechtsnormen vorzugehen. Und wenn die Staatsanwaltschaft der Bundesrepublik nun gegen Mitglieder des Com-

puter Chaos Clubs eruiert, so will er ein Exempel statuieren, das gerade dieses klar machen soll. Sie unterschatzen zudem, sei es aus Naivität oder Mangel an Wissen und Vorstellungskraft, die Schadensmöglichkeiten, die selbst einem noch so vorsichtig geplanten und gut gemeintem Hackerangriff auf Computersysteme innewohnen können. Sie überschätzen oft ihr technisches Wissen und verwechseln ihre wunderbare Trickkiste mit soliden Kenntnissen. Und nicht zuletzt unterschätzen die edlen Hacker die politische und kriminelle Versuchung, die ihr Tun begleitet.

Wie zu Zeiten des Kalten Krieges Volkstanzgruppen und Kulturzirkel gerne von Spionageexperten zu Rekrutierungszwecken missbraucht wurden, so bieten Organisationen von der Art der Hackerclubs heute ideale Rekrutierungsmöglichkeiten für Spionage und Berufsverbrechen aller Art. Experten wundern sich eigentlich, warum internationale Organisationen politischer oder verbrecherischer Art zu Erpressungszwecken und Geldbeschaffung noch nicht mit den Methoden der Hacker operieren. Statt Bomben in Flugzeugen oder Bahnhöfen könnte man auch ausgeklügelte Viren in Lebenswichtigen Rechenzentren plazieren, mit Zeitzünder, versteht sich, und Telefonanruf, dass ein neutralisierendes Programm geliefert würde, falls bestimmte Geld- oder politische Forderungen erfüllt würden bevor der Zeitzünder wirksam wird. Dass das bisher, so scheint es, noch nicht geschehen ist, hat zwei Erklärungen: Zum einen stehen diesen Organisationen offenbar hinreichend ausgebildete Leute nicht zur Verfügung, da echte Experten auch und gerade legal sehr gut verdienen können, und zum zweiten sind solche Virusattacken meist weniger gefährlich, als es dem Laien erscheinen muss. Lebenswichtige System werden meistens gegen allerhand Ausfälle geschützt, und glaubwürdige Angriffsdrohungen sind sehr schwer auszuführen. Trotzdem würde der Experte einen schweren Stand haben, in einer solchen Situation die Entscheidungsträger überzeugen zu wollen, dass sie nicht auf die erpresserische Drohung eingehen müssen.

Die Hacker spielen mit dem Feuer. Dem Staatsanwalt wird alles recht sein, einige von ihnen hinter Gitter zu bringen. Die Hacker wollen, wie das Kind in der Geschichte von des Kaisers neuen Kleidern, die Gesellschaft auf Probleme, Schwächen und ungerechte Informationsverteilung aufmerksam machen. Das mag verdienstvoll sein. Aber sie sind keine Kinder, sondern verantwortbare Erwachsene, und sie müssen, als edle Hacker, ihre Methoden so wählen, dass sie ihre persönlichen Risiken einschätzen können. Der Terrorist weiss, was er riskiert, wenn es schief geht. Der edle Hacker wird es unsanft noch lernen müssen.

Chapter 5

Erfahrung im Zeitalter ihrer technischen Reproduzierbarkeit

(25.4.89)

Programme sind abspielbar, wie Partituren, Schallplatte oder Filme; Und Computer sind ihre Abspielgeräte. Programme sind notierte Handlungsweisen, wiederholbar und modifizierbar. Ihr Abspielen kann abhängig gemacht werden von der Eingabe von zusätzlichen Daten an vorbestimmten Punkten des Handlungsablaufs. Daten, sagen wir, oder besser Zeichen. Daten sind interpretierte Zeichen. Aber die Interpretation der vom Computer abgespielten Handlungsabläufe und vorgeführten Zeichen bleibt dem Betrachter, dem Benutzer überlassen. Die vom Computerprogramm vorgeführten Handlungsweisen manipulieren nur Zeichen, aber eben auch alles, was sich durch Zeichen darstellen lässt. *Bedeutung* erhalten diese Zeichen erst durch den *sozialen Kontext*.

Experten haben *Wissen, Erfahrung und Handfertigkeiten*. *Wissen* gilt als Gemeingut, ist erlernbar und notierbar, und nur Berufsgeheimnisse schränken den freien Austausch von Wissen ein. Wissen wird in Handbüchern, Tafelwerken, Enzyklopaedien und Fachmonographien notiert und tradiert. *Erfahrung* trennt den Experten vom Gelehrten, Erfahrung erwirbt man im Umgang mit der Materie, sie ist meist schwer verbalisierbar und betrifft nicht die Daten sondern den Umgang mit ihnen, das Navigieren in ihnen, das Gespüre. *Handfertigkeit* (oder auch Kopffertigkeit), Routine, Geschicklichkeit betreffen die Geschwindigkeit und Sicherheit im Ausführen. Ein Schachmeister kennt die Schachliteratur (Studien und vormals gespielte Partien), das ist sein Wissen. Er spürt aber auch, welche Partien und Studien in einer gegebenen Situation relevant sind, das ist seine Erfahrung, und er erfasst Situationen schnell und hat schnellen Zugriff zu seinem Wissens- und Erfahrungsschatz, das ist seine Routine. Ein Lawinenexperte unterscheidet sich in dieser Hinsicht kaum vom Schachspieler, sein Schachbrett ist der Lawinenhang, sein Wissen die Physik von Schnee und Eis, seine Erfahrung die Erinnerung an unzähligen Lawinenhänge und Wettersituationen.

Im Computer lassen sich Daten (Wissen) sehr kompakt speichern und sequentiell relativ schnell (Routine) abfragen. Aber das *Ausformulieren von Erfahrung* macht Schwierigkeiten. Diese Schwierigkeit ist dem Begriff Erfahrung inherent: Liesse sich Erfahrung algorithmisch erfassen, so würden wir nicht mehr von Erfahrung, sondern eben von Wissen reden. Das *Reproduzieren von Erfahrung* aber ist möglich: Wir können die Handlungsabläufe speichern, die ein Experte vornimmt, wenn er mit einer computerisierten Wissensbank umgeht. Wir können diese Handlungsabläufe

automatisieren und analysieren, sie selber wieder katalogisieren und das Umgehen mit diesen Katalogen einem ähnlichen Prozess unterwerfen. Die auf dem Computer gespeicherte Wissensbank ist ein Handbuch auf einem Medium mit Erweiterungsmöglichkeiten. Und wie ein gutes Handbuch nicht nur Inhaltsverzeichnis und Index hat, sondern auch Tabellen, Bilder, Kreuzverweise, die aus der Erfahrung hervorgegangen sind, so verfügt eine Wissensbank nicht nur über Abfragemöglichkeiten, sondern eben auch über gespeicherte und reproduzierbare Erfahrung.

Expertensysteme sind Daten- und Wissensbanken, auf denen Erfahrung erst reproduziert, dann modifiziert und schliesslich auch produziert werden kann, sie wachsen durch ihren Gebrauch. Aber, einen Gedankengang Walter Benjamins abwandelnd, die Kunstleistung des Experten wird dem Publikum durch diesen selbst in eigener Person präsentiert, während die Kunstleistung des Expertensystems durch eine Apparatur vermittelt wird. Die Apparatur, die die Leistung des Experten vor das Publikum bringt, ist nicht gehalten, diese Leistung als Totalität zu respektieren. Sie nimmt unter Führung der Programmierer, Designer und Wissensingenieure laufend zu dieser Leistung Stellung. Die Handlungsabläufe, die dem Benützer zur Verfügung gestellt werden, bilden die fertig komponierte Software. Dadurch wird die Leistung des Experten einer Reihe von logischen und algorithmischen Tests unterworfen. Der Experte büsst die Möglichkeit ein, auf den Benützer zu reagieren und sich ihm anzupassen. Der Benützer fühlt sich in den Experten nur soweit ein, als er sich in die Apparatur einfühlt. Er übernimmt also dessen Haltung: er testet. Das ist keine Haltung, der Kultwerte ausgesetzt werden können.

Der Experte ist gerade darin Subjekt, als dass er mit seiner Ehre und oft mit seinem Leben dafür bürgt, dass er sein Wissen und seine Erfahrung verantwortungsvoll einsetzt. Das montierte Expertensystem wird dieser Verantwortung enthoben. Experten können eine Berufshaftpflichtversicherung abschliessen. Für Expertensysteme müsste es demnach eine Produktehaftpflichtversicherung geben. Das hiesse aber, dass der Benützter des Expertensystems seines Subjektes verlustig wird, dass ihm abgesprochen wird, selber, gerade mit dem Expertensystem, neue Erfahrungen zu machen und zu produzieren. So wie ein Arzt nicht seiner Verantwortung enthoben wird, wenn ein Druckfehler in der Arzneimittelbeschreibung Ursache eines Kunstfehlers ist, so kann auch der Benützer von Expertensystemen seiner Verantwortung *nicht* enthoben werden. Expertensysteme sind motorisierte Handbücher für Experten, nicht Roboter, die ungelernte Arbeiter ersetzen.

Chapter 6

Die unsichtbare Software oder der Tod im Lift

(3.6..89)

Für die meisten von uns ist Software, die uns direkt betrifft, entweder etwas, was auf dem PC zu Hause oder im Büro läuft, oder aber der Datenverarbeitung in den verschiedensten Verwaltungen dient. Fehlerhafte Software kann zwar Schaden anrichten, der uns Zeit und Nerven kostet, und manchmal auch Geld. Aber diese sichtbare Software bedroht uns kaum mit dem Tod.

Software spielt auch eine zentrale Rolle in der Raumfahrt, in neueren Flugzeugmodellen wie dem Airbus, in der Flugkontrolle, in Atomkraftwerken, in Werkzeugmaschinen, in modernen medizinischen Geräten, wie dem Tomographen und Bestrahlungsgeräten, in unseren Autos, in Aufzügen.

Dass fehlerhafte Software in Atomkraftwerken oder im Airbus tödliche Folgen haben kann, ist uns allen klar. In diesen Fällen gibt es auch Sicherheitsvorkehrungen, die zwar nicht absolute Sicherheit garantieren, aber dennoch, gemessen an den bekannten Pannen, zeigen, dass das *Missachten dieser Vorkehrungen unverhältnismässig mehr Schaden angerichtet hat*, als fehlerhafte Software selbst. In der Autoindustrie, speziell bei General Motors, werden die in Chips verpackten Programme nicht nur extensiv getestet, sondern auch formal analysiert und verifiziert.

Umsomehr muss es befremden, dass in harmloser erscheinenden Bereichen, konkret im Falle eines Aufzuges, in den USA ein tödlicher Unfall geschehen ist, dessen Ursache, wie es scheint, eindeutig einem Softwarefehler zuzuschreiben ist. Das softwaregesteuerte System, das Türen und Abfahrt des Aufzuges kontrolliert, hat zugelassen, dass ein Kind zwischen Türen eingeklemmt vom abfahrenden Lift grausam zerrissen wurde. Aehnliche Fälle sind in medizinischen Bereichen bekannt. Mindestens zwei Patienten erhielten in den USA tödliche Dosen Bestrahlung, weil die softwaregesteuerten Geräte nicht das taten, was sie laut Spezifikation hätten tun müssen. Dass die beiden hier zitierten Beispiele in den USA geschehen sind, tut nichts zur Sache. Es ist anzunehmen, dass sie auch bei uns geschehen könnten. Es zeigt höchstens, dass die USA problembewusster ist, und dass dort solche Fälle auch an die Oeffentlichkeit kommen.

Ohne hier auf technische Einzelheiten einzugehen, müssen wir uns doch die Frage nach der *Betriebsbewilligung softwaregesteuerter automatisierter Geräte* stellen, und die Frage nach der *Haftpflicht und Versicherbarkeit*. Die Betonung liegt hier bei *softwaregesteuert*. Betriebsbewilligungen für Aufzüge unterstellen die Installation dieser rigorosen Materialkontrollen und erfordern diverse Notvorkehrungen mechanischer Art. Werden solche Notvorkehrungen durch softwaregesteuerte Regler ersetzt, müssten diese eigentlich auch in die Betriebsbewilligung eingeschlossen sein.

Das schliesst formale Spezifikation und Verfikation dieser Software mit ein.

Wer in den oben beschriebenen Fällen haften wird, haben die amerikanischen Gerichte noch nicht entschieden. Aber Fahrlässigkeit der Ingenieure kann nur nachgewiesen werden, wenn Normen der Sorgfaltspflicht existieren, die sowohl von der Gesellschaft gefordert, von den Gerichten anerkannt und von den Fachleuten respektiert werden. Bleibt also die Frage der *Versicherbarkeit softwareerzeugter Risiken*. Würde die Produkthaftpflichtversicherung softwareverursachte Schäden miteinschliessen ? Sicher nicht, ohne eine Risikoanalyse. Softwareverursachte Schäden sind logischer Natur. Im Gegensatz zu Materialschäden, die gewissen statistischen Gesetzen unterliegen, liegt es in der Natur der logischen Fehler, dass sie dem System inhärent sind und sich in gleicher Konfiguration des Systems auch zwangsläufig wiederholen. Solche Risikoanalysen sind durchführbar, aber teuer. Im Falle von Space Shuttle werden sie auch durchgeführt. Aber im Falle eines Aufzuges ist es denkbar, dass das Ersetzen von mechanischen Notvorkehrungen durch softwaregesteuerte Regler prohibitiv verteuert würde, wenn eine zu kleine Anzahl solcher fachgerecht kontrollierter Regler produziert werden sollte.

Automatisieren ist teuer und lohnt nur dort, wo die Widerverwendbarkeit von Prozessen gross genug ist, oder aber dort, wo die Komplexität der Prozesse zwar transparent ist, aber nicht durch Menschen durchgeführt werden kann. Das Einstellen einer Waschmaschine mit unabhängiger Temperaturregelung und Waschgangwahl sollte auch einen Hausmann nicht überfordern. Das Betreiben von fünf Aufzügen in einem öffentlichen Verwaltungs- oder Bürohochhaus erfordert automatisierte Steuerung. Man kann sich leicht vorstellen, was alles passieren kann, wenn der Aufzug nur gerufen werden kann, wenn er frei ist, und nur ein Stockwerk aufs Mal als Fahrziel angegeben werden kann. Die alten Paternoster Aufzüge lösten das Problem auf rein mechanische Weise: sie fuhren ständig, mit vielen kleinen Kabinen, wie eine Art Riesenrad. Die Passagiere mussten aufspringen. Aber ihre Geschwindigkeit war gerade deswegen beschränkt. Aufzüge mit Vorbestellungsmechanismen lassen sich gerne von Kindern missbrauchen. Die Spezifikation einer automatischen Steuerung ist eine beliebte Testaufgabe in Hochschulkursen zum parallelen Echtzeitprogrammieren. Die dabei anfallenden Probleme sind für den Laien eher komplex, aber sie sind von Fachleuten hinreichend gut verstanden, sodass man eine korrekte Spezifikation und Implementierung eigentlich zur Sorgfaltspflicht von Computer- und Softwareingenieuren zählen können sollte.

Es ist Zeit, über diese Fragen vermehrt nachzudenken und danach zu handeln. Und es lohnt sich im Kleinen anzufangen, nach dem Tod eines unschuldigen Mädchens in einem falsch programmierten Lift, und nicht erst nach Katastrophen wie Tschernobyl.

Chapter 7

Software:

Produkt, Diensleistung oder Medium

(August 1989)

7.1 Computer kaufen, wozu ?

Neulich fragten mich die Leiter eines Kleinbetriebes, ob ich ihnen bei der Computerisierung ihres Ladens helfen könnte. Aber ihre Fragen betrafen nur die Auswahl der Maschinen. Ich erwiderte ihnen, dass mich die Art ihrer Fragen etwas erstaune. Sie wür den mir vorkommen wie die Fragen eines Unmusikalischen, der sich endlich mit Musik beschäftigen wollte und nun nach geeigneten Geräten Ausschau hielte, verunsichert ob Grammaphon, CD-player, Radio, Klavier oder Synthesizer, besonders auch welche speziellen Marken und Produkte in dieser Palette, ihm den besten Weg zur Musik würden weisen können. Dass Musik und Musik ganz verschiedene Dinge bezeichnen können, schien irrelevant bei dieser Frage, so wie bei ihnen die Frage, wozu und was denn mit den Computern gemacht werden sollte.

Diese Situation ist, im Kleinen wie im Grossen, leider nicht untypisch. Computerisierung wird leider immer noch identifiziert mit der Beschaffung von Maschinen, ohne dass die Software, die auf den Maschinen laufen soll, ins Zentrum der Betrachtung gestellt würde. Vom Personal, das notwendig ist, diese Software fachgerecht zu warten, ganz zu schweigen. Man spricht dann von Softwarekrise, wenn es nicht klappt. Aber niemand würde von einer Musikkrise sprechen, wenn die verschiedensten Geräte eben nur das spielten, was die Gerätefabrikanten als Demonstrationsmusik vorbereitet hätten.

Aber was charakterisiert dann nützliche und gute Software, respektive was macht denn Aerger, wenn Software nicht die *erhofften*, doch *nicht explizit bewussten Erwartungen* der Käufer und Benützer erfüllt?

7.2 Software als Fertigprodukt ?

Software ist eine *strukturierte Sammlung von Programmen*, die aus einem *universalen Computer* eine für speziellere Zwecke geeignete Maschine machen. Diese speziellere Maschine dient der automatischen Ausführung von Arbeitsgängen, der Automatiserung von bestimmten Tätigkeiten. Es ist davon auszugehen, dass im Normalfall diese Tätigkeit interaktiv mit dem Benützer zusammen ausgeführt wird. Die Abstände zwischen den Interventionen des Benützers bestimmen den Grad der Interaktivität. Aktivitäten, die so automatisert werden variieren sehr. Sie

reichen vom einfachen Rechnen und Buchhalten bis zur Robotersteürung in Fertigungsabteilungen grosser Betriebe. Die Struktur der Interaktivität nenne ich *Dialog*, die Ausgestaltung des Dialogs, *Interface.* Betreffen die Interventionen des Benützers nur die Eingabe von Daten oder die Wahl von Optionen, so sprechen wir von Benützersoftware.

In diesem strengen Sinne erscheint Benützersoftware selten fertig auf dem Markt. Taschenrechner-Emulatoren, Terminal-Emulatoren, Kommunikationsprogramme, Verarbeitungsprogramme für elektronische Post, Texteditoren und Druckerprogramme gehören in diese Kategorie, aber auch Computerspiele. Software, die zur Herstellung und Konfiguration von anderer Software dienen, zählen wir hier nicht dazu. Auch die in nicht als Computer empfundenen Geräten versteckte Software nicht, sei es in medizinischen Geräten, in Aufzügen, in Autos oder in Industriemaschinen und Robotern. Um die Diskussion möglichst anschaulich zu halten, wählen wir die Beispiele vorwiegend aus dem Bereich der Büroautomatisierung im weitesten Sinne. Aber das meiste gilt auch für die versteckte Software, vielleicht in sogar grösserem Masse.

Das Spezifizieren solcher Benützerprogramme ist zwar nicht ganz so einfach, wie es scheinen könnte, aber die Arbeitsabläufe, die automatisert werden, sind meist mehr oder weniger klar. Problem entstehen oft erst durch unbedachte Anhäufung von Optionen und Erweiterungen. Der Appetit kommt mit dem Essen, und die Lust auf Mehr an automatiserten Komfort kommt beim Benützen der Software. Das verleitet uns, diese Benützerprogramme mit Optionen zu versehen, die das Verändern dieser Programme miterlaubt, sei es durch eigenes Konfigurieren oder gar Programmieren von Erweiterungen. Und damit begeben wir uns auch auf das Glatteis.

7.3 Betriebssysteme für spezielle Anwendungen

Betriebssyteme sind letztlich auch Programme, aber sie dienen nicht der Ausführung bestimmter wohldefinierter Endhandlungen, sondern derer Organisation. Betriebssyteme verhalten sich zu den Benützerprogrammen wie die Verwaltungsburokratie zu den von der Verwaltung offerierten Dienstleistungen. Aber wie komplexe Dienstleistungen nicht ohne Organisation angeboten werden können, so ist das Handhaben eines Computers ohne Betriebssystem nicht möglich, oder doch äusserst mühsam und kontraproduktiv. Erst das Betriebssytem macht den Computer zu dem was wir von ihm sehen, spüren, womit wir umgehen und woran wir verzweifeln können. Ein *benützbarer* Computer ist immer eine *Maschine mit einem Betriebssystem*, etwas was man beim Diskutieren von Vor- und Nachteilen von IBM, Digital, Sun (Commodore, Atari, McIntosh) nicht genug betonen kann.

Die Sache wird dadurch noch weiter kompliziert und verworren, als man Interfaces und Betriebssysteme auf andere Interfaces und Betriebssysteme *aufpfropfen* kann, dass man sie *simulieren* und, wie man bei absolut identischer Verhaltensweise sagt, *emulieren* kann. MS-DOS ist nur für Tastaturen geschrieben, und mit dem Erweitern durch Fenster (Windows) und Mäusen wird es immer mehr dem Betriebssytem der Macintosh-Welt angenähert. Die Uebergänge werden immer fliessender, und Betriebssyteme entwickeln Varianten und Akzente wie Sprachen in einem Immigrantenstaat.

Aber was die Sache mit der Software noch komplizierter macht, ist die Verwischung der Grenzen zwischen Benützersoftware und Betriebssystemen. Die meisten Spreadsheet-Programme und Datenbank-Softwarepakete sind aufgepfropften Betriebsystemen mit speziellen Umfeldanpassungen ähnlicher, als laufsicherer wohldefinierter Benützersoftware. Sie bestehen aus Konventionen und Formaten, Fileverwaltungen, Editoren und Programmiersprachen zum Ausprogrammieren von Op-

tionen, Makros, Konfigurationen, Interfaces und Dialogen. Sie sind in ihren Anwendungsmöglichkeiten so universell (und unspezifisch), dass Experten von Nöten sind, um sie in Benützersoftware zu verwandeln. Was man mit solchen Softwarepaketen kauft, ist weder ein Produkt, noch eine wohldefinierte Dienstleistung, sondern ein *Milieu*, eine *Subkultur*, in welcher die erwünschte Benützersoftware dann erst erträumt, entworfen, ausformuliert und schliesslich ausprogrammiert werden soll. Ob diese Subkulturen nun Expertensysteme, Künstliche Intelligenz, Büroautomatisierung, Spreadsheet, Datenbanken, Funktionelles oder Objektorientiertes Porgrammieren heissen, ändert daran wenig. Und in diesem Sinne kann man von diesen Softwarepaketen auch als von einem *Medium* sprechen.

7.4 Zumutbarkeit und Sorgfaltspflicht

Benützersoftware von der Stange ist selten wohldefiniert, selbst wenn die primäre Anwendung klar erscheint. Textverarbeitungssysteme zum Beispiel bestehen aus Editoren, Formatierprogrammen, Wortsuchmechanismen, Rechtschreibekontrollen und Druckprogrammen. Selten wird bei der Beschreibung dargestellt, wie diese Programmteile logisch und verhaltensmässig zusammenhängen. Die Kunst des Schreibens von Benützeranleitungen steckt noch in den Kinderschuhen. Zudem weist der Autor oder die Vertriebsgesellschaft jede Verantwortung für das tatsächliche oder vermutete Verhalten des Programmpakets vehement von sich, nicht im Kleingedruckten Teil des Kaufvertrages, sondern in grossen Lettern, lautstark und selbstbewusst. Dass eine solche Zurückweisung vor Gericht nur beschränkt Gültigkeit haben würde, ändert daran nichts. Sie genügt, die Benützer vor gerichtlichen Schritten abzuhalten, da sie versprechen, allfällige Prozesse hinreichend zu komplizieren. Die Konsumenten solcher Programme lassen sich ja ohnehin das meiste gefallen, auch wenn sie sich gerne über den Zustand der Dinge verbal beklagen. Was früher das Wetter oder die Politik für das allgemeine Klagebedürfnis der Leute bedeutete, erbringt heute das Leiden mit der Software. Mann und Frau erleiden sie einzeln und kollektiv, aber schliesslich schwimmt man im Strom der Modernisierung und Automation. Dabei könnte man, falls es sich nicht um unrechtmässig erworbene, auf dem grauen Markt kopierte Software handelte, durchaus etwas tun. Aber dafür wären kollektive Schutzverhatensmuster notwendig, wie Kosumentenschutz, Selbsthilfegruppen, gewerkschaftliche Interventionen. Im Beispiel der Textverarbeitung könnten minimale Forderungen an die Qualität der Büroautomatiserung im Gesamtarbeitsvertrag der Betroffenen mitformuliert werden, statt dass die Beschwörungsformel der Arbeitsplatzerhaltung nur zur Leugnung der eigentlichen Probleme aufgerufen würde. Vieles, was als Software angepriesen wird, automatisiert Arbeitsprozesse zu schnell und zu unbedacht, auf unzumutbare Weise. Der Begriff der *Sorgfaltspflicht* bei der Automatiserung ist noch nicht einmal in Sichtweite.

7.5 Konfigurieren und Programmieren

Benützersoftware von der Stange in unserem eng definierten Sinne ist, wie schon gesagt, eher selten. Häufiger ist der Fall, dass Softwarepakete für speziellere Anwendungen konfiguriert und ausprogrammiert werden müssen. Oft handelt es sich dabei auch um grundsätzliche Probleme der *Konzeption von automatiserbaren Handlungsabläufen.* Datenbankdesign und Dialogdesign sind dabei vielleicht die herausstechendsten Beispiele. Ich habe noch kein automatisiertes Katalogsystem in Grossbibliotheken oder Dokumentationszentren angetroffen, das meinen Wunschvorstellungen nur annähernd entsprochen hätte. Bei den meisten solchen Automatisierungslösungen

sticht hervor, dass vieles, was von Hand früher zwar mühsam aber möglich war, jetzt unmöglich geworden ist. Zentrale Ursache solchen Ungenügens ist das Problem der Kommunikation zwischen dem potentiellen Anwender und dem Softwareingenieur.

Es wird gemeinhin unterschätzt, dass beim Automatisieren von Handlungsabläufen, das heisst, beim Uebersetzen von der Erfahrung, die diesen Handlungsabläfen zu grunde liegt, in funktionstüchtige Softwaresysteme, diese Erfahrung erst bewusst gemacht, dann konzeptualisert und schliesslich verbalisert werden muss. Würde diese Arbeit mit aller *Sorgfalt des Gesunden Menschenverstandes* ausgeführt, so erwiese sich, dass das Programmieren letztlich der einfachste Schritt in der Automatisierung wäre. Aber diese vom Gesunden Menschenverstand geleitete Kommunikation zwischen denen, die die Handlungsabläufe vor der Automatiserung durchschaut haben und denen, die sie automatisieren und codieren müssen, erforderte von beiden Parteien nicht nur Gesunden Menschenverstand, sondern viel *Zeit, Einfühlungsvermögen und Fantasie* für die Möglichkeiten und die Gefahren der neuen Medien, realistische *Bescheidenheit* in der Vorgabe des zu Ereichenden, eine *Bauhaus-ähnliche Disziplin* in der Bewahrung von Funktionalitäten und zugleich eine fast anthroposophische *Achtung für Handwerklichkeit* und menschliche Ergonomie. Wissenschaft ist nichts als *selbstbewusster gesunder Menschenverstand*, sagte einer der fürenden Logiker dieses Jahrhunderts (Quine), und das trifft ganz besonders auf die Ingenieurwissenschaften zu.

Gute Software entsteht nur auf diesem Hintergrund. Schlechten Softwarelösungen sind fast immer Verletzungen dieser anspruchsvollen Sorgfaltspflicht vorausgegangen, weder die Produktionsbedingungen noch die Ausbildung der Fachleute erlauben im allgemeinen solche hohen Ansprüche. Wenn lebenswichtige Prozesse unter unfreundlichen Bedingungen automatisiert werden, sind Aengste zu Recht am Platz. Aber Aengste lähmen nur unsere kritischen Fähigkeiten und leisten Mythenbildung Vorschub.

7.6 Staatlich vereidigte Software Auditoren

Software ist manchmal Produkt, manchmal automatiserte Dienstleistung, manchmal auch nur ein Medium, indem Erfahrung vermittelt und reproduziert werden kann. Diese Ambivalenz erschwert den Umgang mit der Software. Verantwortlichkeit, Haftpflicht und Sorgfaltspflicht werden zwischen den Produzenten und Endbenützern der Software hin- und hergeschoben. Die Produzenten, besser Autoren, der verschiedenen Softwareteile und der zu ihrer Herstellung verwendeten Softwareinstrumente können sich nur schlecht untereinander abgrenzen. Wer haftet letztlich für die Folgeschäden fahrlässiger Programmierung ? Wieweit kann in solchen Haftpflichtfragen Regress genommen werden ? Wieweit reicht die vernünftigerweise vorauszusettzende Sorgfaltspflicht ? Wer soll als Schiedsgericht in solchen Fragen auftreten ? Wer soll Betriebsbewilligungen für softwaregesteuerte Anlagen erteilen, wer für öffentlichrechtliche Datenbanken ? Wieweit sollen solche Betriebsbewilligungen den Betreiber von seiner Haftpflicht entlasten ?

Es gibt Gerichtsurteile, die den Autor eines medizinischen Handbuches für die Folgeschäden einer durch einen *Druckfehler* entstandenen Fehldosierung eines Medikaments verantwortlich machen. Es gibt ein Gerichtsurteil, in dem eine Automechanikerwerkstätte für den Folgeschaden einer fahrlässigen Revision eines etwas komplizierteren Automodells verantwortlich gemacht wurde mit der Begründung, dass in diesem Fall dem Endbenützer das Kontrollieren der Revision *wegen der Komplexität des Gerätes* nicht mehr zugemutet werden konnte. Beide Fälle sollten die Softwarehersteller aufschrecken und die Beruforganisationen der Informatikberufe mobilisieren. Die Versicherungsgesellschaften sollten überhäuft werden mit Anträgen für Berufs- und Produkthaftpflicht versicherungen im Softwarebereich. Und die

Versicherungsgesellschaften sollten solchen Anträgen mit grösster Vorsicht, aber nicht ablehnend begegnen. Wenn das Publikum in Sachen Softwarefehler endlich prozessfreudig und die Gerichte zu Entscheidungen gezwungen würden, erst dann enstünde der soziokulturelle Druck, der nötig ist, damit Selbstdisziplinierungsmassnahmen der Verantwortungsträger im Softwarebereich in Gang gesetzt würden. Die Berufsorganisationen müssten den *staatlich vereidigten Softwareauditor* etablieren, die Versicherungen müssten ihr Risk-Engineering auf Softwareprodukte ausweiten und der Gesetzgeber müsste sich mit dem Stand der Technologie Software auseinandersetzen. Softwareengineering als Disziplin mit wissenschaftlicher Grundlagen hat mehr zu bieten, als man sich allgemeinhin zugestehen will, auch wenn noch viele Probleme offen sind. Aber wir alle, Informatikexperten, Informatikverwalter, Informatikanwender, Informatikopfer, lassen uns fast alles gefallen, was uns an Pfuscherei und Schlendrian zugemutet wird. Sorgfaltspflicht betrifft alle, Produzenten, Abnehmer und Benützer. Der Umgang mit softwaregesteuerten Geräten ist nicht nur problematisch, weil die Materie neu und komplex ist, sondern weil die sozio-kulturellen Bedingungen für ihre Handhabe den technischen Möglichkeiten weit hintan sind.

Chapter 8

Cantus Firmus oder wie erfinde ich mein Passwort

(9.9.89)

Folgende Geschichte ist vielleicht wahr oder aber gut erfunden. Eines morgens, vielleicht war es schon Frühling, kam Bill Hacker, wie schon öfters, spät zur Arbeit. Alle seine Kollegen sassen vertieft über ihren Schreibtischen und grübelten offenbar mit Bleistift und Papier an einem Problem, das von Ferne einem Kreuzworträtsel zu gleichen schien. Bill fand ein Dokument auf seinem Schreibtisch, das mit DRINGEND rot markiert war. "Passwortalarm" stand gross darauf, kleiner darunter "Aus Sicherheitsgründen müssen alle Computerbenützer ihr Passwort sofort ändern". Es folgten acht Seiten mit Richtlinien, wie das neue Passwort beschaffen sein sollte. Es sollte mindestens zwölf Zeichen enthalten. Es durfte auf keinen Fall irgendwo schriftlich oder im Computer festgehalten werden. Es sollte nicht nur aus Zahlen oder nur aus Buchstaben bestehen. Es sollte kein Folgen von Buchstaben enthalten, die als Wörter im Duden, Webster oder Larousse zu finden waren. Es sollte keine Folgen von zwei oder mehr aufeinander folgenden gleichen Zeichen enthalten. Es sollte keine Folgen von Zeichen enthalten, die als Folgen von benachbarten Zeichen auf der Tastatur zu finden war. Es sollte keine üblichen Vor- oder Nachnamen enthalten..... Das Dokument enthielt acht Seiten solcher Ausschlussregeln. Bill Hacker war verwirrt. Er nahm Bleistift und Papier und fing an zu pröbeln, bevor er ans Ende des Dokumentes angelangt war. Nach ein paar Stunden hörte er plötzlich Lachen und Fluchen. Die Kollegen zeigten auf die letzte Zeile im zweiten Abschnitt der vorletzten Seite. "Das einzige Passwort, das alle diese Bedingungen erfüllt, ist 1;LirpA;9891". Es wahr schon Frühling, es war der erste April.

Zwei Dinge sind sicher wahr an dieser Geschichte: Dass wir kaum Spezifikationen in Ruhe zu Ende lesen und dann erste noch über die Konsequenzen nachdenken und dass unsere Leichtsinnigkeit in der Passwortwahl grenzenlos ist. Zum letzteren erschien kürzlich im angesehenen britischen Guardian eine Reihe von Empfehlungen, wie Passwörter zu wählen sind, um die Sicherheit von Computer drastisch zu erhöhen, ohne dass teures Geld investiert werden müsse. Die Empfehlungen waren auf den ersten Blick überzeugend, bei genauerem Hinsehen aber nur kurzfristig erfolgversprechend. Die Hauptidee war das Vermischen von den früher üblichen Passwörtern. Statt Heidi oder Franz dann halt Heanz oder Fraidi; statt London oder Zürich halt Zondrich. Ein solches verfahren vermindert die Wahrscheinlichkeit, dass ich ein Passwort erraten kann tatsächlich erheblich, aber für automatisierte Ratealgorithmen nicht genug. Alle anderen Nachlässigkeiten im Umgang mit Passwörtern werden dabei nicht berührt.

Gute Passwörter sind für ihren Besitzer leicht erinnerlich, für andere aber schwer zu erraten. Das hört sich klar an und scheint leicht verständlich. Aber beide Bedingungen sind nicht wohldefiniert. Das erste hat mit der subjektiven Assoziationsfähigkeit und Selbstdisziplin des Passwortbesitzers zu tun, das zweite mit den möglichen Ratemethoden, die automatisert werden können. Das explizite Auschliessen von Ratemethoden erinnert an die barocke Disziplin des Cantus Firmus, dem Schreiben von vierstimmigen kurzen Harmoniesequenzen, in denen "zu einfache" Tonprogressionen und -wiederholungen verboten waren. Cantus Firmus ist allerdings keine Kompositionsmethode, sondern diente der Uebung der Erfindungsgabe und Schärfung der Aufmerksamkeit geduldiger Kompositionsschüler. Auch unter den bösen Hackern gibt es geduldige Schüler.

Persönliche Passwörter sind immer relativ kurz. Man stelle sich vor, was passieren würde, wenn der PIN-Code 24 Zeichen enthielte. Aber in der elektronischen Kommunikation zwischen Maschinen sind Passwörter von erheblicher Länge nicht nur möglich, sondern erforderlich. Sie müssen dann auch nicht leicht erinnerlich sein, sondern leicht konstruierbar.

Die Grundlagenforschung in der Kryptographie beschäftigt sich seit längerem damit, leicht kodierbare, aber schwer dekodierbare Kryptosysteme zu entwickeln. Sie hat dabei erstaunliche Resultate vorzuweisen, auch wenn noch viele Fragen offen sind. Allerdings haben die Amerikanischen Behörden die besten Lösungsvorschläge verboten, weil damit die vom Geheimdienst geforderte Abhörbarkeit von Datenleitungen gefährdet würde. Ein altes Gesetz verbietet das chiffrierte Uebertragen von Daten auf öffentlichen Datenträgern. Dass jeder Text Chiffre eines anderen Textes sein kann, ist dabei schon den alten Gesetzgebern entgangen.

Damit kommen wir zum Problem der Spezifikation und deren impliziten Konsequenzen zurück. Ein gutes Passwort oder eine gute Chiffriermethode bietet offenbar in der Vorstellung dieser Behörde zwar leichtes Kodieren für den Freund, unmögliches Dekodieren für den Feind aber auch bequemes wenn auch nicht zu leichtes Dekodieren für die eh nur freundlich gesinnten, unparteiischen Ueberwachungsorgane.

Ob die Verantwortlichen der Amerikanischen Behörden ihren eigenen Forderungskatalog gelesen und gar dann noch überdacht haben, bleibt zu bezweifeln.

Chapter 9

Programmieren im WK

(21.9.89)

Es ist unvermeidlich, Büro-Automatiserung nimmt ihren Einzug auch in der Schweizer Armee. Allerdings nicht, wie man denken könnte, von höchster Stelle konzipiert und befohlen, sondern spontan da und dort, wo die Initiative einzelner Kommandanten und Adjutanten solches vorantreibt. Im WK begegne ich solchen Projekten, und die militärische Geheimhaltung und Takt verhindern, dass ich über meine konkrete Erfahrung berichte. Aber soviel sei gesagt, die Arbeit, die wir, meine Kameraden und ich, in diesen WK's leisten, ist den Umständen entsprechend mehr als erstaunlich gut, alle sind motiviert und machen letztlich aus Enthusiasmus sogar Ueberstunden, bereiten sich zu Hause auf die Arbeit vor und freuen sich, etwas zu machen, was bleibende Wirkung hat, statt in Schiessübungen auf den Alpen die Subvention der Bergbauern in Form von Militärentschädigungen zu erhöhen.

In einer Milizarmee, die so radikal jeden Soldaten auch als Kämpfer sieht, ist wenig Zeit für professionelle Arbeit im Umfeld Logistik und Technik. Alle müssen in der wenigen Zeit, die zur Verfügung steht, ihre Kampfbereitschaft aufrechterhalten. So mindestens lautet das Dogma unserer, durch die Armeeabschaffungsinitiative in Frage gestellten, Schweizer Armee. Im kriegsgeplagten Israel gibt es auch nur eine Milizarmee, eine allerdings im Aktivdienst stehende. Nur, darauf will ich hinaus, ist der Prozentsatz der kampfbereiten Truppe an der gesamten Armee erstaunlich klein. Logistik, Technik und Bürokratie, nehmen einen enormen Platz ein. In der Schweiz wird dieser Teil vorwiegend durch zivile Bundesangestellte abgedeckt. Die Armeeverwaltung ist eine eigene Behörde, und die Haupleute und noch höheren Offiziere machen ihren Verwaltungskram selbst oder im Sekretariat ihres Arbeitsgebers. Für die verschiedenen Grosscomputersysteme stehen zivile Mathematiker und Programmierer zur Verfügung. Aber für die kleinen, kostengünstigen Automatiserungsprojekte ist keine einzige stetige Arbeitskraft vorgesehen. WK-Soldaten entwickeln Prototypen, testen und dokumentieren, soweit wie unter diesen Umständen möglich, die Software, die adhoc zusammenspezifiziert wurde. Im WK-Prammierdienst entstehen Lösungen für bestimmte Automatiserungsaufgaben, deren Funktionieren nur gewährleistet ist, solange die Leute damit umgehen, die auch bei deren Entwicklung dabei waren. Werden diese Leute aber befördert, umgeteilt oder haben sie ihre Dienstpflicht beendet, so wird auch die Software, die sie entwickelt haben, aus dem Dienst genommen werden müssen. Dokumentation und Wartungsprotokolle sind im allgemeinen, falls überhaupt vorhanden, so knapp abgefasst, dass sie für die Nachbenützer kaum verwendbar sind. Viele Fehler, Bugs, sind nach wie vor vorhanden und deren Vermeidung und Umgehungsstrategien waren Geheimnisse, die nur in mündlicher Ueberlieferung tradiert wurden. An Portabilität solcher Lösungen, oder gar Freigabe zum Gebrauch in anderen Einheiten darf nicht gedacht werden.

Dreissig Mann im WK, das sind neunzig Wochen, also doch zwei Mann-Jahre

(Ferien und andere Ausfälle miteingerechnet). Wenn man die Zeit abzieht, die erforderlich ist, um die Kommunikation zwischen den dreissig Mann zu gewährleisten, bleibt ein halbes Mann-Jahr übrig. Was in einer solchen Einheit tatasächlich geleistet wird, entspricht diesem Kalkül. Die Software-Lösungen, die da produziert werden, sind kaum schlechter, als was erreicht würde, hätte man sie auf dem freien Markt, nicht gekauft, fertige Lösungen dieser Art existieren kaum, sondern bestellt. Die meisten Mängel, oh ja, es gibt deren viele, sind nicht dem WK-Ambiente zuzuschreiben, sondern Misskonzeptionen, die auch in der Privatwirtschaft grassieren: Dass die zukünftigen Benützter am Entwicklungsprozess nicht teilnehmen müssen, dass der Forderungskatalog des Bestellers in sich fehlerfrei ist, dass man wichtige Entscheidungen in der Gestaltung der Benützerschnittstelle dem Programmierer überlassen kann, dass die Forderungen an das zu entwickelnde Programm, einmal formuliert, unveränderlich sind, wie Spezifikation einer Brücke, dass Benützerdokumentation mit der linken Hand im Nachhinein geschrieben werden kann. Wir können hier den ganzen Katalog der marktüblichen Pfuschereien durchgehen, dass es im WK nicht besser aussehen kann, als im ach so professionellen freien Markt, ist uns allen klar.

Dennoch könnte vieles besser aussehen, wenn das Konzept des Programmiersoldaten etwas gründlicher durchdacht würde. Man könnte zum Beispiel statt sechs mal drei Wochen, einmal vier Monate WK leisten. Aber vorallem müssten die Offiziere und Adjutanten, die mit der in der Armee erstellten Software umgehen, sich mit den Tücken dieses Tuns in einer Milizarmee vertraut machen können. Das wichtigste aber wäre, dass die, die solche Software entwickeln lassen, nicht nur dass Wissen und die Erfahrung in solchen Dingen haben, sondern auch die Zeit, die es vom Besteller erfordert, solche Automatiserungsaufgaben bis zur Lösung voranzutreiben. Software ist keine Sofortware, schon gar nicht, wenn sie unter WK-Bedingungen erstellt werden muss.

Chapter 10

Software-Zertifizierung ?

(15.11.89)

Die Schweizerische Treuhand- und Revisionskammer hat vor nicht zu langer Zeit in einer Fachmitteilung Leitlinien veröffentlicht, die der Abgabe von *Softwarezertifikaten* zugrunde gelegt werden sollen. Zertifiziert wird hier Software, die in den Buchhaltungen grösserer Betriebe Verwendung findet und vom Rechnungsprüfer mit in Betracht gezogen werden muss. Damit masst sich die Schweizerische Treuhand- und Revisionskammer etwas an, dem sie nicht gewachsen sein kann. Sie gibt vor, durch ihr Softwarezertifikat einen Qualitätsausweis auszugeben, der letztlich keiner ist.

Das vorgeschlagene Softwarezertifikat gleicht mehr einer Checkliste für eine journalistische Bewertung der zu prüfenden Software, vergleichbar den Leitlinien, die ein Autojournalist befolgen sollte, wenn er einen Fahrtest für ein neu auf dem Markt erschienenes Auto durchführen und den er dann in der Autobeilage einer soliden Konsumentenzeitschrift beschreiben sollte. Aber im Gegensatz zum Autojournalisten, der sich immerhin auf universelle, vom breiten Publikum akzeptierte Kriterien für die Qualitäten eines Autos stützen kann, gibt es im Softwarebereich nicht einmal diese. Die Disziplin der Softwareprüfung ist zwar weit entwickelt, aber selbst under den Spezialisten fü Anwendersoftware zu wenig bekannt.

Eine Buchhaltungssoftware automatisiert buchhalterische Operationen und Hilfoperationen und fügt diese zu einem Ganzen zusammen. Dabei muss diese Software den Anforderungen genügen, die von Steuerbehörden und Gesetzgeber erhoben werden. Das betrifft vor allem die Formate der Buchhaltung, die internen Kontrollsysteme und Rekonstruierbarkeit der verbuchten Operationen. Zudem muss sie alle Operationen, nicht nur die arithmetischen, korrekt ausführen. Erfahrungen mit Computerkriminalität zeigen, dass gerade beim Automatisieren der Buchhaltung grosse Möglichkeiten bestehen, sich den üblichen Kontrollmechanismen gerade dadurch zu entziehen, dass man vortäuscht, diese Kontrollen in die automatisierte Buchhaltung eingebaut zu haben.

Alle Anforderungen an eine Buchhaltungssoftware gehen ein in den Anforderungskatalog, der der Softwarespezifikation vorausgeht. Der Lebenszyklus eines Softwarepaketes erstreckt sich aber weiter: Anforderungskatalog, Spezifikation, Implementation, Revision, neue Anforderungen, Neuspezifikation, Neuimplementation, Benützerkritik, neue Anforderungen, etc. Das Testen und Evaluieren von Software ist inzwischen zu einer Disziplin herangewachsen, die ihre eigenen Standards hat und sich durchaus mit der Disziplin der Materialtests vergleichen lässt, obwohl noch keine gesetzlich bindende Normen festgelegt worden sind. In dieser Disziplin bedeutet *Zertifizieren* das extensive Testen einer schon Verifizierten und Validierten Software unter den echten Bedingungen der späteren Anwender. Eine Software gilt als verifiziert, wenn sie vom Herstellerteam und von einem unabhängigen Team am Herstellerort sys-

tematisch kontrolliert, getestet und für korrekt befunden wurde. Sie gilt als validiert, wenn sie und ihre Dokumentation bei privilegierten Probekunden vom Herstellerteam und einem unabhängigen Tester geprüft und auch vom Kunden akzeptiert wurde. Die Zertifizierung stützt sich zwangsläufig auf diese Testergebnisse ab.

Was wird denn überhaupt in diesem, von der Schweizerischen Treuhand- und Revisionskammer vorgeschlagenen Softwarezertifikat geprüft? Die Leitlinien bestehen aus einem sechs-seitigen Text, einer Checkliste und einigen Zertifikatsbeispielen. Sie betreffen vier Bereiche: Verarbeitungsregeln, interne Kontrolle, Dokumentation und den Bereich Ausgabe, Aufbewahrung und Nachprüfbarkeit. Unter den Grenzen der Zertifizierung wird vermerkt, dass nur die Software selbst hinsichtlich ordnungsgemässer Buchführung beurteilt wird, nicht aber die Rahmenorganisation, nicht die Richtigkeit der Daten und auch nicht die richtige Bedingung der Programme durch die Benützer. Es wird weiterhin vermerkt, dass die Softwareprüfung eine Momentaufnahme ist, die sich nur auf eine bestimmte Version der Software auf einer bestimmten Maschinenkonfiguration bezieht.

Die Leitlinien nehmen keinen Bezug auf den Lebenszyklus der Software und auf die verschiedenen Dokumentatiosebenen, die diesen Lebenszyklus begleiten. Was geprüft wird, ist ob die Software den Anschein macht, als ob der gesetzliche Anforderungskatalog an die Buchführung in die Implementation umgesetzt worden sei. Der Bezug auf die Versionsnummer und die Maschinenkonfiguration ist zwar richtig, aber er suggeriert hier Gründlichkeit, die in einer zweiseitigen Bescheinigung gar nicht gegeben werden kann. Die Checkliste, die den Leitlinien beigegeben ist, vermischt Punkte kleinsten Details mit den schwierigtstenFragen der Softwarebeurteilung. Es ist leider klar, dass bei ihrer Ausarbeitung kein professioneller Software-Ingenieur mitgearbeitet hat. Ich unterstelle der Schweizerischen Treuhand- und Revisionskammer guten Willen und echte Sorge bei der Bemühung um ein Softwarezertifikat. Aber das Resultat ist mehr geeignet, meine Befürchtungen zu schüren, als mein Vertrauen zu stärken. Mit diesen Leitlinien werden aber jene beruhigt, die aufschrecken sollten. Das einzige Verdienst der Schweizerischen Treuhand- und Revisionskammer in Sachen Softwarezertifikat besteht darin, das Thema aufgebracht zu haben. Jetzt müssen Bücherexperten, Softwareexperten, Juristen und staatliche Stellen zusammenarbeiten, um die Bedingungen für ein sinnvolles Zertifikat erst zu ergründen.