Informatik als Handwerk, Technik, Wissenschaft [1]

Roland Vollmar

Lehrstuhl Informatik für Ingenieure und Naturwissenschaftler
Universität Karlsruhe (TH)
Am Fasanengarten 5
D-76128 Karlsruhe
vollmar@ira.uka.de

Zusammenfassung: Das Ineinandergreifen wissenschaftlicher, technischer und handwerklicher Vorgehensweisen - in unterschiedlichem Grade - prägen die Informatik.
Diese Sicht auf die Informatik soll dazu beitragen, die Stärken der jeweiligen Ansätze zu entwickeln und die ihnen eigenen Schwächen zu vermeiden.

Computerentwicklung und Handwerk

Wenn sich auch die Informatik als Fach erst etwa 20 Jahre nach der Funktionsfähigkeit der ersten vollautomatischen, programmgesteuerten und frei programmierbaren Rechnern etablierte, bilden doch diese ihre Basis.
Entwicklung und Bau der ersten Maschinen und auch die Konzeptionen Babbages waren handwerklich geprägt, wenn „Handwerk“ in der umgangssprachlichen Weise verstanden wird [2].
Charles Babbage, der Mitte des 19. Jahrhunderts in seinem Entwurf der "Analytical Engine" die Konzeption der späteren vollautomatischen Rechner vorwegnahm, war von einer erstaunlichen handwerklichen und künstlerischen Begabung - wie schon ein Blick auf Zeichnungen von ihm beweist. Mit Clement als Mechaniker arbeitete er eng zusammen, scheiterte letztlich aber an der zeitbedingten mangelnden Fähigkeit zur Fertigung hochpräziser Mechanik (und an persönlichen Eigenheiten). Ein Sprung um fast 100 Jahre führt zum ersten von dem großen deutschen Computerpionier Konrad Zuse gebauten Automaten, der Z1, den er zwischen 1936 und 1938 zusammen mit Freunden realisierte. Erstaunlicherweise war es ebenfalls noch ein mechanisches Gebilde. Durch die geniale Idee, eine binär arbeitende Maschine zu bauen, konnte er auf die mechanisch aufwendig zu fertigenden Staffelwalzen, Sprossenräder oder Zahnräder, wie sie in den mechanischen Rechenmaschinen seit Schickard und auch in der Konzeption von Babbage verwandt wurden, verzichten und mit gelochten Blechen auskommen. Diese wurden in unterschiedlicher Version sowohl für den zuerst (1936) fertig gestellten Speicher als auch für Steuer-

[1] Sehr viel ausführlicher ist das Thema in [V03] behandelt.
[2] Die Encyclopaedia Britannica [EB99] verzichtet auf eine Definition von Handwerk, wohingegen in einem alten Lexikon [BL55] eine zu finden ist: „Handwerk, Gegenstände mittels einer durch mehrjährige Ausbildung [...] erlangten Fertigkeit herzustellen, bei d. man sich zwar bestimmter Werkzeuge bedient, aber die Handarbeit die Hauptrolle spielt und die Maschine in den Hintergrund tritt.“

und Rechenwerk verwandt. Die Konzeption der Maschine erwies sich als korrekt, sie arbeitete aber wegen leicht vorkommender Verkantungen unzuverlässig. Bei der Entwicklung der Z1 ist das Handwerkliche nicht nur beim exakten Herstellen Tausender Bleche zum Ausdruck gekommen, sondern viel beeindruckender in der Konzeption, die in der besten Tradition der Handwerkskunst stand, wie sie z.B. im 18. Jahrhundert durch John Harrison, den Konstrukteur mehrerer Uhren, die das Längengradproblem lösen halfen, exemplifiziert werden kann. Damit war aber die handwerkliche Entwicklung noch nicht am Ende. Statt auf die weiteren Zuseschen Maschine Z3 und Z4 einzugehen, sei F.J. Dyson [DF00] zur amerikanischen Situation zitiert: "Der Bau von Computern begann als handwerkliche Tätigkeit; John von Neumann leitete eine wilde Truppe von Mathematikern und Ingenieuren am Institute for Advanced Study in Princeton, und ähnliche Gruppen bauten anderswo ähnliche Maschinen. In seinem Buch *The Soul of a New Machine* beschreibt Tracy Kidder den Geist einer solchen handwerklichen Industrie, die bis in unsere Zeit überlebt hat. Aber die von Kidder beschriebene Maschine und das Ingenieurteam, das sie entwickelte, haben nicht überlebt. [...] Einige Jahre später gestanden Seymour Cray und Danny Hillis ihr Scheitern ein; sie waren die letzten jenes Schlags unabhängiger Unternehmer, die Computer im handwerklichen Stil produzierten. Sie konnten sich auf dem Markt nicht gegen die Großunternehmen behaupten. In der Computerbranche geht die Zeit der handwerklichen Produktion dem Ende entgegen." Bemerkenswert ist dabei, dass nicht *ein* Handwerk bestimmend war, sondern Mechanik, Elektrik, Elektronik Grundlage waren. (Bei Cray und Hillis ist wohl „handwerklicher Stil" eher metaphorisch aufzufassen, da sie ungewöhnliche, „individuelle" Architekturen entwarfen und realisierten.) Diese letzte Satz des Dysonschen Zitats bezieht sich lediglich auf den Bau von Computern. Dass er nicht für die gesamte Informatik gilt, wird später dargelegt werden.

Computer als universelle Maschinen

Computer sind im Gegensatz zu anderen technischen Artefakten universell einsetzbar [3] ; dazu mussten nicht nur Methoden und Ergebnisse der Mathematik, Physik und Elektrotechnik übernommen und adaptiert werden, sondern es entstanden neuartige Fragestellungen, zu deren Beantwortung ein eigenes Methodengebäude entwickelt werden musste, das die Basis eines eigenen Fachgebietes bildet. (Der Unterschied zur Autoproduktion und -verwendung ist in [V03] detailliert erläutert.) Beispielhaft werden die durch den Computereinsatz bewirkten Änderungen in Wissenschaft, Wirtschaft und täglichem Leben skizziert.

Änderungen in der Wissenschaft

John von Neumann sieht schon 1954, als er sich bereits etwa zehn Jahre mit Rechnern und ihren Anwendungen befasst hatte, einen Einsatzbereich, der über die arithmetischen Fähigkeiten hinausgeht: "[...] der Zweck einer Rechenmaschine [ist] lediglich, eine

[3] Darauf und nicht auf den in der Informatik gebräuchlichen Terminus nimmt die Überschrift Bezug.

menschliche Tätigkeit, die man selbstverständlich auch ohne maschinelle Hilfe durchführen könnte, nämlich das Lösen von mathematischen Problemen durch Rechnen, zu beschleunigen [...]. Es handelt sich dabei großenteils um das Auflösen von Problemen, die entweder aus der reinen Mathematik oder aus der angewandten Mathematik oder aus den angrenzenden Wissensgebieten - Physik, Chemie usw. - herrühren, wobei das Problem unter Umständen gar nicht ein Rechenproblem ist." [vN63] Er deutet dabei auf eine höchst bedeutsame Nutzung des Rechners hin, nämlich *Modelle* zu bearbeiten, sowohl Modelle physischer Strukturen oder Prozesse als auch solche wirtschaftlicher Abläufe oder abstrakt gegebener Systeme. V. Cherniavsky identifizierte gar die Informatik mit "interpretierbarer Modellierung" (und betonte die Sichtweise „Virtualität“ (s. z.B. [S01])).
Ein anderer, für die Durchsetzung der Rechner entscheidender Punkt ist der des "Hoffähigmachens" numerischer Lösungen - im Gegensatz zu "geschlossenen Lösungen" der (vor-maschinellen) Mathematik. Auch dieser Aspekt wird schon von John von Neumann deutlich gesehen.
Aber es werden nicht "nur" numerische Lösungen akzeptiert in Fällen, in denen analytische nicht erreichbar sind, sondern es wurden auch Probleme mit exakten Methoden behandelbar, z.B. im Zusammenhang mit Matrizen, an deren Lösung in der vor-maschinellen Zeit ihrer schieren Größe wegen nicht zu denken war - natürlich trug auch das rapide steigende Speichervolumen dazu bei -. So sind z.B. Gleichungssysteme (sogar nichtlineare) mit mehreren Millionen von Unbekannten, wie sie z.B. im Maschinenbau bei Finite-Element-Methoden auftreten, lösbar geworden (s. z.B. [O97]).
Die Verfügbarkeit von Rechnern führte zu grundlegenden Änderungen in der Methodik der Angewandten Mathematik und hatte z.B. den Einsatz von Monte-Carlo-Methoden und von probabilistischen Algorithmen zur Folge. F. J. Dyson [DF00] nennt noch ein Beispiel: "Die wichtigsten unter den neuen Instrumenten, die Wolszczans Entdeckung ermöglichten, waren Computerprogramme. Der technische Fortschritt im Bereich der Astronomie hat heute mehr mit Software als mit Hardware zu tun." [4]
Generell lässt sich für die Natur- und Ingenieurwissenschaften feststellen, dass sie durch die Nutzung von Computern in Geräten, zur Steuerung von Experimenten, zur Auswertung experimenteller Daten und nicht zuletzt durch Simulationen und Berechnungen von Modellen nicht nur eine Änderung erfahren haben, sondern entscheidende Schritte voran zu machen in der Lage waren. Ähnliches gilt auch für andere Wissenschaften, z.B. die medizinische Forschung.

Änderungen in der Wirtschaft

Die Nutzung von Rechenanlagen in der Wirtschaft ergab sich fast zwangsläufig: Große und mittlere Unternehmen waren an den Einsatz von Lochkartenmaschinen gewöhnt (und auch darauf angewiesen), und so verlangten die größeren Möglichkeiten, die sich durch

[4] Generell wird dies in [M719] unter dem Begriff „Technik“ angesprochen: „... Die verbreitetste Einschätzung des Verhältnisses von T. und Wissenschaft (v.a. Naturwissenschaft) ist naturalistisch ausgerichtet. Sie nimmt an, der Mensch mache sich mit Hilfe der T. die Natur bzw. ihre Gesetze nutzbar, indem er wissenschaftliches, d.h. theoretisches Wissen anwendet. Diese Auffassung verkennt, dass die modernen Erfahrungswissenschaften in ihren Forschungsmöglichkeiten vom augenblicklichen Stand der Beobachtungs-, Experimentier- und Messtechnik sowie der Informations- und anderer T.en abhängen. Selbst Naturforschung im engsten Sinn (z.B. in der Astronomie und in der Biologie) hängt in ihren Resultaten immer auch vom Stand der Beobachtungs-T. ab (z.B. vom Auflösungsvermögen der Fernrohre oder Mikroskope). ...“

ihren Anschluss an Rechenautomaten boten, kein entscheidendes Umdenken. Zunächst stand die Automatisierung der Buchhaltung, der Lohnabrechnung und dann des gesamten Bestell- und Rechnungswesens an, wobei *ein* zentraler Rechner das Herz bildete. Aber auch in diesem Umfeld erwiesen sich die Möglichkeiten der Modellbildung und der Verwendung von Modellen im Computer als höchst hilfreich bei der Integration der einzelnen Abläufe, so dass insbesondere mit dem Aufkommen von PCs und lokalen Netzen auf dieses Hilfsmittel nicht mehr verzichtet werden kann. Neben dieser konsequenten Übernahme von Geschäftsabläufen werden Computer auch für neuartige Aufgaben eingesetzt. Die Einbeziehung von Methoden des Operations Research und der Prozessautomatisierung führte zu immer mächtigeren Systemen, wobei auch solche zu nennen sind, die zur Neuentwicklung höchst komplexer technischer Produkte durch vernetzte, global positionierte Teams beitragen.

Änderungen in Arbeitswelt und technischer Umwelt

Hinter dem Begriff der "Prozessautomatisierung" steckt ebenfalls eine ungeheuer erfolgreiche Entwicklung, bei der der Rechnereinsatz das integrative Element zwischen Elektrotechnik, Maschinenbau und Informatik darstellt.
Nach mechanischer Regelung z.B. bei Wasserspeichern oder der Dampfmaschine wurden zunächst spezielle elektrische Regler eingesetzt. Rechner, insbesondere in ihrer geringfügig modifizierten Form als Prozessrechner, erlaubten vor allem durch ihre Geschwindigkeit, ihr Speichervolumen und die Möglichkeit nichtlinearer Regelung, einen wesentlichen Sprung. Sie werden zur Steuerung und Regelung von Produktionsabläufen in praktisch allen Bereichen eingesetzt: Mit der Fähigkeit der Computer zur Bildverarbeitung und Mustererkennung erweiterten sich die Nutzungsfelder, wobei vor allem auch im militärischen Bereich spektakuläre Anwendungen zu finden sind [5]. U.a. sind Chiffrieren und Dechiffrieren, wie überhaupt die modernen Anwendungen der Kryptographie ohne Rechner nicht denkbar.
Weniger spektakulär, aber von größerem Einfluss auf unser tägliches Leben sind die Rechner in den sog. "eingebetteten Systemen". In Haushalts- und Bürogeräten, in der Unterhaltungselektronik, in Geräten zur Unterstützung bei Behinderungen oder auch in Autos ist eine große Zahl von Rechnern vorhanden, ohne dass sie sichtbar oder sonst bewusst wahrnehmbar würden.

Informatik als Wissenschaft

Schlagwortartig seien die wichtigsten Grundbegriffe genannt, die mit zur neuen Methodologie der Informatik beitragen [6]:
1) *Algorithmisierung*: Das Auffinden und Erfinden von Algorithmen ist nicht nur die Grundlage zur Lösung von Problemen, sondern verhilft zum tieferen Verständnis der verschiedensten Phänomene und Prozesse [7].

---

[5] T. Sasaki [S03] stellt die Frage, ob nicht durch den Einsatz von Computern in der Waffentechnik seit etwa zehn Jahren das Führen eines Krieges auch für demokratische Staatswesen weniger tabuisiert wurde.
[6] Ein Teil der Aspekte ist detailliert in [G97] behandelt.

2) *Formalisierung*: Formale Beschreibungen und Symbolmanipulierungssysteme erweisen sich bei der Spezifikation realer Systeme und ihrer Verhaltensweisen als außerordentlich wichtig.
3) *Virtualisierung:* Die Möglichkeit, auf einem beliebigen (Universal-)Rechner die Arbeitsweise eines beliebigen anderen nachvollziehen (simulieren) zu können – und dies, falls gewünscht, über mehrere Stufen – („virtuelle Maschinen“) und Modelle (realer oder irrealer) Systeme im Rechner darzustellen und agieren zu lassen („virtuelle Realität“) erleichtert und beschleunigt Entwicklungsarbeiten resp. ermöglicht ein (ungefährliches) Trainieren in (potentiell) risikoreicher Umgebung [8].
4) *Komplexitätsuntersuchungen*: Die Einsicht, dass formale Objekte und Prozesse eine inhärente Komplexität besitzen, zu deren Messung unterschiedliche Ressourcenmaße herangezogen werden können, trägt wesentlich zu ihrem Verständnis bei und hat weitreichende Auswirkungen bez. ihrer Anwendbarkeit.
5) *Untersuchung komplexer Systeme*: Während lange Zeit in den Naturwissenschaften Fortschritte dadurch erzielt wurden, dass Grundphänomene isoliert untersucht wurden, erlauben Informatikmethoden das Herangehen an interagierende, stark miteinander verkoppelte Systeme. Mit den Computern hat die Informatik ganz wesentlich mit höchst komplexen, nämlich chaotischen Systemen zu tun. Wie Hartmanis [H95] sagt, ist die Berechnungen ausführende Hardware universell und chaotisch: Die kleinsten Veränderungen der Befehle oder der Daten können beliebig große Differenzen der Resultate nach sich ziehen [9].
Abstrahierungen von realen Systemen erlauben eine Modellbildung, die sich in Programme übersetzen lässt und durch deren (wiederholte und parametrisierte) Ausführung Rückschlüsse auf die realen Systeme gezogen und diesen innewohnende Gesetze aufgedeckt werden können. (Noch) nicht existierende Systeme können untersucht werden, wobei sogar in sog. virtuellen Welten die physikalischen und/oder biologischen Gesetzmäßigkeiten modifiziert werden können. Beschleunigung oder Verlangsamung von Vorgängen ist ebenso möglich wie z.B. deren Umkehrung.
Ergebnisse solcher *Simulationen* sind, wie auch die von umfangreichen Berechnungen, z.B. der Wettervorhersage, oder der statistischen Analyse sehr großer Datenmengen nur dann nutzbar, wenn sie in einer dem Menschen entsprechenden Form vorliegen. Dies gewährleisten Werkzeuge zur *Visualisierung* bzw. im dynamischen Fall, z.B. bei Crashtests, der *Animation*.
Hiermit ist nur ein kleiner Teil von dem angesprochen, was das informatische Vorgehen zu einer neuen Methodologie werden ließ. In den Natur- und Ingenieurwissenschaften bildet es u.a. nach Hartmanis und Lin [H92] neben theoretischem und experimentellem Vorgehen die dritte Säule der wissenschaftlichen Arbeitsweise. Die Informatik erweitert die durch Theorie und Experiment gebotenen Möglichkeiten beträchtlich, insbesondere in den bisher nicht zugänglichen Bereichen komplexer Systeme, wofür hier die Telekommunikationsnetze als *ein* Beispiel angeführt seien. Komplexe Vorgänge werden verstehbarer, es können Voraussagen über ihr (künftiges) Verhalten gemacht werden, die

---

[7] Die herausragende Bedeutung der Algorithmisierung wird z.B. in [G98] betont: „Another possible name for the field, which is not intended to cover the full scope of CS but, rather, its heart and basis, is ‚algorithmics'.“
[8] Als Beispiele seien für den ersten Fall das Ausführen von Mehrprozessoralgorithmen auf einem Einprozessorsystem und für den zweiten Flugzeug- und Fahrzeugsimulatoren erwähnt.
[9] Man denke z.B. nur an eine Vorzeichenumkehr oder an eine Endlosschleife anstelle eines definierten Resultates.

auch dazu benutzt werden können, entsprechende physikalische, chemische, biologische und technische Prozesse zu optimieren.

Hardware und Software unter handwerklichen und technischen Aspekten[10]

Computer stellen aus verschiedenen Winkeln betrachtet scheinbar sehr unterschiedliche Geräte dar (darauf wird näher in [V202]eingegangen). Dies gilt auch für Computer im (schulischen) Unterricht: „Der Computer nimmt in der Ausbildung verschiedene Rollen ein. Zum ersten ist er selber Gegenstand des Unterrichtes, zum zweiten ein Werkzeug im Arbeitsalltag und zum dritten ein Medium zur Über- und Vermittlung von Informationen." [H02]
Die "universelle" Fähigkeit des Computers ist seine Stärke, aber auch in gewisser Weise seine Schwäche: Sie macht es erforderlich, ihn an die gewünschte Aufgabenstellung anzupassen, sprich, ihn zu programmieren. Auch dies ist nichts prinzipiell Neues: “Long before the word *software* came into general use, there was a need to describe longer, complicated sequences of calculation. For this purpose, it was patent to support the write-up of intermediate results on paper by forms indicating the course of calculation. [...]. Konrad Zuse, in 1935, was lead by a blank of this kind used in calculating the superposition of two rectangular moment areas to his first ideas about a formula-controlled computer [...]." [B00]
Für die frühen Computer wurde die Programmerstellung (bis in die sechziger Jahre) noch nicht als problematisch angesehen, einfach deshalb, weil der wissenschaftlich ausgebildete Personenkreis, dem sie zugänglich waren, mit den Algorithmen auch die Programme entwarf bzw. gewisse Standardroutinen von den Rechnerentwicklern kostenlos mitgeliefert erhielt.
Der Handwerk- und Technikcharakter der Informatik tritt vor allem im Softwarebereich zutage: Ebenso wie in manchen „klassischen“ Handwerken, z.B. im Schmiede- oder Uhrmacherhandwerk werden in der Informatik die benötigten Werkzeuge selbst hergestellt und das richtige Umgehen mit ihnen wird vermittelt [11]. In ihr werden nicht nur die Rechner konzipiert und zumindest viele Vorbereitungen zu ihrem "Bau" getroffen, sondern sie ist auch zuständig für das Wissen um die Möglichkeiten und die Grenzen ihres Einsatzes, und nicht zuletzt lehrt sie die Methoden, um das "Universalwerkzeug Computer" durch Programmierung dem jeweiligen Problem anzupassen. Zemanek wies 1993 darauf hin, dass in diesem Bereich noch „viel zu tun ist“, was nicht der Software-Industrie allein

[10] Dass das gleichzeitige Nennen von Handwerk und Technik als problematisch angesehen werden kann, lässt sich aus dem folgenden Zitat ablesen [F94]: „ ... war die ursprüngliche Konzeption und die künstlerische Richtung des Bauhauses weitgehend durch die Tradition William Morris’ bestimmt, die eine Rückkehr zum Handwerk als Reaktion gegen die moderne Technologie fordert.“

[11] Für diese Auffassung kann ich z.B. auch Brooks [Br96] als Kronzeuge anführen, obwohl ich seine Charakterisierung für zu eng halte: „....the computer scientist is a *toolsmith* – no more, but no less. It is an honourable calling.“

überlassen werden könne [Z93] [12] – und er stellt dabei auch den Bezug zum Handwerk her. Jeder Nutzer eines Computers wird wohl dieser Aussage auch heute noch zustimmen.
Im Gegensatz zu anderen Ingenieurwissenschaften ging die Informatik nicht aus *einem* Handwerk hervor. Dies mag auch *ein* Grund dafür gewesen sein, dass in ihr lange Zeit Ausbildungsberufe auf unterschiedlichen Ebenen kaum angeboten bzw. gefragt waren, was ihr sicherlich nicht zum Vorteil gereichte, allein schon deshalb, weil vieles von „Dilettanten“ erledigt werden musste und weil „handwerklich Begabten“ keine Perspektiven eröffnet wurden. Eine weitere Ursache mag darin gelegen haben, dass der Umgang mit Computern in deren Jugendjahren fast ausschließlich Mathematikern und Physikern oblag, deren Tätigkeit gemeinhin gleichsam als „Geheimwissenschaft“ betrachtet wurde; ihre Vermittlung wurde weder als wünschenswert noch als leicht möglich angesehen. Die Haltung zur Programmierung änderte sich mit dem oben beschriebenen Einsatz des Computers in immer neuen Gebieten, und mit zunehmender Komplexität der Anwendungen wuchsen die Programme. Das spiegelt sich u.a. in der Wortwahl wider: D. Knuth nannte sein berühmtes Werk „The Art of Computer Programming“ [K76]. Wenn die Bezeichnung wohl auch metaphorisch zu verstehen ist, so wurde es doch bald offensichtlich, dass Programmieren zu einer "Technik" (bzw. einer „Wissenschaft“ [Gr81]) entwickelt werden musste. Jetzt wird sie mit dem Begriff "Software Engineering" bezeichnet.
Nach F.J. Dyson [DF00] ist die Software-Industrie eine handwerklich geprägte Industrie: "Die Computer-Software entstand als Hilfsmittel der Wissenschaft und breitete sich später auf alle Bereiche der Industrie und des Handels aus. Überall, wo ernsthaft mit Computern gearbeitet wurde, lernten junge Menschen, Programme zu schreiben und anzuwenden. Trotz Microsoft und anderer Großunternehmen ist die Software-Industrie bis heute in weiten Teilen handwerklich strukturiert. Wegen der gewaltigen Vielfalt spezieller Anwendungen wird es stets Raum für Einzelne geben, die auf der Grundlage ihres ganz besonderen Wissens Software entwickeln. Es wird stets Marktnischen geben, in denen kleine Software-Hersteller überleben können. Das Handwerk der Software-Entwicklung wird nicht aussterben. Und das Gewerbe der kreativen Anwendung von Software wird

[12] „Hier möchte ich einen Exkurs in die Gegenwart machen und einen Aspekt der Computertechnik anpeilen, der vielleicht auch heute wieder Universitätsaufgabe sein könnte: Der Aspekt der Programmierwerkzeuge, der Softwarefabrik. Während nämlich die Hardware alle Vorteile der Kombination von Physik, Elektronik und Fertigungstechnik zu äußerst preiswerter Herstellung ausnützt, wird die Software handwerklich gefertigt, aber nicht mit den Vorteilen des Handwerks, vom persönlichen Kontakt auszugehen und die Erfahrung für die Wünsche und Nöte des Benützers einsetzen zu können; und weil Software-Fertigung so mathematisch aussieht, nimmt man sie auch als Tummelplatz von Genies und Quasigenies hin. Man redet zwar von Software Engineering (man darf nicht übersehen, dass dies ein von Mathematikern geprägter und initiierter Ausdruck ist), aber von einer Produktionsstrecke in Analogie zur Hardware wird noch lange keine Rede sein können. Ohne auf Einzelheiten einzugehen: Werkshallen und erst recht Produktionsstrecken beziehen Effektivität und Verlässlichkeit aus den verwendeten Werkzeugmaschinen. Für die Software gibt es zwar den Ausdruck "tools", aber keine universell befriedigende Werkzeugmaschinen der Programmierung. Alle Versuche, Softwarefabriken in Gang zu nehmen, sind gescheitert [...]. In der Software-Produktion fehlt der Informationstechnik noch ein wesentliches Element: Die beherrschte Beziehung zwischen Werkzeug und Produkt. Es klafft eine Lücke mit teuren Folgen infolge von Unverlässlichkeit, Einzelfertigungspreis, improvisierter Dokumentation und krassen Liefertermin-Überschreitungen.
Vielleicht wäre hier die Rückkehr zu vom kommerziellen Druck freien Universitätsvorgangsweisen der rechte Weg - die akademische Durchuntersuchung des Software-Herstellungsvorgangs auf seine Automatisierbarkeit. [...] Man müsste den Entwurfs- und Herstellungsvorgang bis in die Grundlagen analysieren und langfristig daran arbeiten - was sich die Industrie tatsächlich nicht leisten kann.“

sogar noch stärker florieren als deren Entwicklung. Die Vielfalt der Programme und die Fähigkeit, sie einzusetzen, haben ein ganzes Spektrum handwerklich strukturierter Industrien hervorgebracht, vom Desktop-Publishing bis hin zur computergestützten Konstruktion und Produktion." [13]
Mit der Ad-hoc-Fertigung von Programmen als Mittel, den Computer in die Lage zu versetzen, einzelne Algorithmen "auszuführen", ist es aber bei wohl allen - zumindest nichtwissenschaftlichen - Anwendungen nicht getan.

Ohne das Handwerk abwerten zu wollen, bin ich überzeugt, dass sich die „handwerklich strukturierte“ Software-Industrie (-und diese Einschätzung Dysons teile ich hinsichtlich der derzeitigen Situation) in eine „technisch orientierte“ wandeln muss. Zumindest müssen aber die bewährten Vorgehensweisen des Handwerks und seine Ethik auch für diesen Bereich übernommen werden. Ein gutes Beispiel dafür sind die „Entwurfsmuster“ für Software, die in Analogie zu den im Handwerk – für Schlosser, Weber bis zu Turmuhren- und Orgelbauer und in der Bildenden Kunst – bekannten Musterbüchern zu sehen sind (von den bekanntesten, nämlich solchen für Köche einmal ganz abgesehen).
Eine solche Entwicklung wird allein schon der Masse der zu produzierenden *qualitativ hochwertigen* Software wegen notwendig sein. Der Bedarf an *solcher* Software scheint mir immer noch größer als dass er mit dem vorhandenen Personal realisiert werden könnte.
Es darf nicht sein, dass jeder, der programmieren kann, etwas „Neues“ „zusammenbastelt“ und dies als ein Produkt „verkauft“. Und auch der Einsatz von Werkzeugen („case tools“) hat beträchtliche Nachteile: Verhindert er doch, dass Software-Erstellung von der Pike auf gelernt werden muss. Ähnlich wie jeder Maschinenbaustudent früher im Praktikum beim Feilen einen sinnlichen Eindruck vom Material gewinnen musste, sollte im Informatik-Studium auf eine „programming literacy“ Wert gelegt werden. Der Mangel daran hat ähnliche Konsequenzen, wie die, die in Klagen von Bauingenieuren gehörte: Das Handhabungswissen von Programmsystemen zum Entwurf auch komplexer Bauten sei relativ einfach zu vermitteln; was fehle, sei die Erfahrung, die nur bei längerem „training on the job“ erworben werde und deren Fehlen verhindere, dass (für Geübte offensichtliche) Schwachpunkte erkannt würden.
Andererseits ist der Aufbau von Software-Systemen, die größenordnungsmäßig eine Million Zeilen Code enthalten, in hinreichender Güte nicht ohne verbesserte Werkzeuge und ohne Computerhilfe zu bewerkstelligen. Ich hege allerdings Zweifel daran, dass dabei die Automatisierung so weit reichen kann, wie dies z.B. beim Compilerbau der Fall ist, nicht nur weil die Komplexität solcher Systeme inhärent größer ist, sondern weil ja in den meisten Anwendungen die menschlichen Nutzer mit ihren divergierenden Ansichten und Bedürfnissen, vielleicht auch denen, zumindest Grundstrukturen der Software zu verstehen und deren Grenzen verdeutlicht zu erhalten, einbezogen werden müssen.

[13] Dass aus der Wissenschaft Anforderungen an das Handwerk gestellt werden, ist natürlich auch nichts Neues. Exemplarisch stehe dafür das Folgende: „Von den angewandten mathematischen Wissenschaften verlangte besonders die Astronomie, die unter Männern wie Bernhard Walther, Johann Werner und Regiomontan in Nürnberg eine hohe Blüte erreichte, feinste Instrumente, die geschickte Schlosser und Schmiede anfertigten. Ebenso war es mit feinen Messinstrumenten, die von den Zirkelschmieden hergestellt wurden, und exakten Waagen, die aus der Werkstatt der Waagemacher kamen.“ [S77]

Nach Tauber [T01] dienen in industrieller Software "die Hälfte bis Dreiviertel der Programmzeilen nicht dem eigentlichen Anwendungsproblem, sondern der Robustheit und Stabilität, der komfortablen Benutzbarkeit und der sicheren Betreibbarkeit." [14] Letztere zu erhöhen und zumindest Software für sicherheitskritische Anwendungen zu verifizieren, ist eine Forderung an die Informatik als *Wissenschaft*, wobei nicht unterschlagen sei, dass halb-automatische Beweiser in relativ kleinen Programmen schon erfolgreich eingesetzt werden. In anderen Bereichen des Software Engineering ist die Informatik von ihrer *technischen Seite* her am Zuge.
Handwerke wirken aufeinander und auf Technik und Wissenschaft ein. So nutzten z.B. Kanonenmacher das Know-how der Glockengießer, nach der *Encyclopaedia Britannica* hatten die aus der Entwicklung der Windmühlen gewonnenen Kenntnisse über Mechanik u.a. Einfluss auf die Herstellung von Uhren, und das Schmiedehandwerk förderte das Aufkommen der Metallurgie.
Im Bereich der Software-Erstellung werden am ehesten die Wechselwirkungen von handwerklichem Vorgehen und wissenschaftlichen Erkenntnissen deutlich. Aber auch auf dem Gebiet der Automatentheorie und der Formalen Sprachen lässt sich beobachten, wie Resultate der Forschung in die informatische Technik übernommen werden. In [H01] ist dazu Folgendes bemerkt: "While applications of automata and language theory to compilers are now so well understood that they are normally covered in a compiler course, there are a variety of more recent uses, including model-checking algorithms to verify protocols and document-description languages that are patterned on context-free grammars."

Der Versuch einer Trennung der Informatik in technische und wissenschaftliche Bereiche ist nicht erfolgreich vorzunehmen: Charakterisiert man (wohl zu naiv) die Technik als „durch Erfindungen geprägt“ und ordnet man der Wissenschaft „Entdeckungen“ zu, so hat man einerseits die Schwierigkeit, dass die Wirkbereiche dieser beiden Begriffe umstritten sind [15], und dass man zum andern daran zweifeln kann, dass es in der Informatik überhaupt Entdeckungen gibt – und damit ein Widerspruch zu oben Gesagtem aufträte. Das konstituierende Element der Informatik, „der“ Computer, ist sicherlich nicht entdeckt worden. (Folgt man Daumas [D75], dann müsste ergründet werden, ob er eine Erfindung oder eine Innovation darstellt.) Entdeckungen der Informatik wird man am ehesten in der Theorie erwarten. Ist aber z.B. die Feststellung, dass unentscheidbare Probleme existieren, eine Entdeckung? Ihre Existenz hängt ja davon ab, ob man die Adäquatheit der Explikation des Berechenbarkeitsbegriffs (z.B. durch die Turing-Maschine) akzeptiert. Aussagen der Theorie beruhen auf einem „gesetzten“ Axiomensystem, also auf etwas, auf das man sich nach Heisenberg [W832] „nur“ einigen kann. In den Naturwissenschaften liegt eine andere Situation vor, aber auch dort muss man vorsichtig sein: Die „Ent-

[14] Nach eigener Erfahrung früherer Programmfertigungen würde es mich nicht wundern, wenn auch in kommerzieller Software ein Teil nie angesprochen wird, entweder weil man Erweiterungen geplant, dann aber nicht ausgeführt hat oder weil man z.B. zwei verschiedene Algorithmen zur Lösung eines Problems implementierte und sich nicht definitiv entschließen konnte, einem den Vorzug zu geben. Damit scheint eine ähnliche Situation vorzuliegen wie in der Biologie: „...erweist sich der allergrößte Teil unseres vorgeblichen Erbgutes aus solcher Sicht eher als Erbmüll, in dem zwar vermutlich, wie in den Abfallhaufen der Vorzeit, in denen Archäologen so gerne wühlen, allerhand Information über die biogenetische Vorgeschichte des Menschen steckt“ [M01]
[15] S. z.B. [D75]: „Die langen Kontroversen über die Bedeutung der Worte Entdeckung und Erfindung sind bekannt. Vor langer Zeit begonnen, sind sie auch heute noch nicht abgeschlossen.“

deckung“ der meisten Transurane z.B. kommt erst durch ihre „Konstruktion“ zustande, da sie auf der Erde nicht (mehr) vorkommen.

Technik und vor allem Handwerk berücksichtigen bei ihren Hervorbringungen ästhetische Ansprüche. Deshalb wird im folgenden gezeigt, dass dies auch für die Informatik gilt und sie sich auch unter diesem Aspekt entsprechend einordnen lässt [16]. Folgte man Duddeck [Du01], so wäre (auch) die Informatik als „Technikwissenschaft“ eine Kunst [17] - und damit, zumindest wenn man die klassische im Auge hat, mit Ästhetik zu assoziieren. In der Encyclopedia Britannica wird ausgeführt, dass oft die Grenze zwischen Wissenschaft und Kunst nicht klar zu ziehen sei [18]. Nach Goethe ist „Kunst: eine andere Natur, auch geheimnisvoll, aber verständlicher; denn sie entspringt aus dem Verstande” [Go98], woraus zu folgern ist, dass sich Wissenschaft und Kunst – zumindest nicht von vornherein – auszuschließen brauchen.
Es dürfte klar sein, dass Personen, die in der Informatik tätig sind, *daneben* aber als Künstler (im üblichen Sinne, d.h. als Komponisten, Maler, ...) wirken als Zeugen für „Ästhetik in der Informatik“ nicht herangezogen werden dürfen. Auch Werke, bei deren *Produktion* Computer eingesetzt wurden, sollen nicht als Beleg dienen. Ebenso wenig darf eine Selbstaussage, wie sie z.B. im Buchtitel „The Art of Computer Programming“ [K76] oder anderen ähnlichen gegeben ist, als Beweis genutzt werden, womit nicht gesagt sei, dass der Bereich der Programmierung außer Acht gelassen werden dürfe.
Nach einem Blick auf die „Definition“ von Kunst in [EB99] [19] werde das „Hervorbringen ästhetischer Objekte“ in der Informatik diskutiert. Nicht-funktional bedingte Ästhetik, wie sie z.B. im (damals modernen) Teakholz-Gehäuse der TR4 oder der Rauchglas-Eleganz der CM2 zum Ausdruck kam, bleibe unberücksichtigt. In der Informatik spielen sowohl beim Erstellen von Modellen als auch beim Entwerfen von Algorithmen und Schaltungen wie auch beim Fertigen von Programmen - wobei diese Aufzählung nicht als erschöpfend anzusehen ist [20]- ästhetische Gesichtspunkte eine Rolle.
Natürlich lässt sich über die Ästhetik von Objekten trefflich streiten; in allen Kunstgattungen sind viele Beispiele zu finden, deren Ein- und Wertschätzung sich im Laufe der Zeit geändert haben – und zwar in beide Richtungen – und solche, die sogar zu *einem* Zeitpunkt auch unter Kennern massive Ablehnung und begeisterte Zustimmung fanden.

---

[16] Auf die Bedeutung einer nicht einseitigen Sicht (hier in Bezug auf das Uhrmacherhandwerk) wird in [J77] hingewiesen: „Man tut gut daran, zu erkennen, dass Zeitmessung sowohl eine Kunst als auch eine Wissenschaft ist, und für ein richtiges Verständnis ist es wichtig, sie als beides gebührend zu würdigen.“

[17] „Technik ist das griechische Techne, die Kunst, das Können, die Geschicklichkeit. [...] Bis in die Renaissance hinein waren alle Techniken Künste: Die Kunst des Wasserbaus, des Ackerbaus, des Schmiedens, die Kunst des Brückenbaus, die Bergbaukünste.“

[18] “[…] whether one considers architecture, medicine, chess, or cookery, there is no clear frontier where the realm of **science** ends and that of **art** begins: the creative person is a citizen of both. “[EB99]

[19] “Art, the use of skill and imagination in the creation of aesthetic objects, environments, or experiences that can be shared with others. The term art may also designate one of a number of modes of expression conventionally categorized by the medium utilized or the form of the product; thus we speak of painting, sculpture, filmmaking, music,...”

[20] So fiel mir zum ersten Mal eine ausserinformatische Wertung in dem Buchtitel „Jewels of Formal Language Theory“ [S81] auf.

Der Einfachheit der Betrachtung wegen sei im folgenden „ästhetisch“ und „schön“ synonym gebraucht. Damit ist noch nicht viel gewonnen, da „Schönheit“ philosophisch gesehen nicht leichter handhabbar ist [21], aber den Vorteil hat, intuitiv einsichtiger zu sein.
Die folgenden Einschätzungen sind natürlich höchst subjektiv, sie sollen aber auch nur beispielhaft zeigen, in welchem Sinne „Schönheit“ verstanden wird.
In der Mathematik hält man häufig kürzere Beweise für schöner als längere (für dieselbe Aussage) [22] [23]. Für Programme kann dies in einer ersten Näherung ebenso gelten, allerdings darf dabei die Kürze nicht mit Unstrukturiertheit erkauft sein. Als unschön sind sicherlich im allgem. solche Programme anzusehen, die sich „entwickelt haben“ (oft nur ein euphemistischer Ausdruck für „zusammengeschustert werden“), d.h. ohne Gesamtkonzeption erstellt und nach und nach erweitert und modifiziert wurden [24] [25]. Für sie gilt wohl das von Markl [M01] über die Erbsubstanz Gesagte.
Bei Algorithmen hängt die Schönheit auch von der Formulierung ab, die geprägt werden kann durch die Möglichkeiten der verwendeten Programmiersprache: So sind rekursive Algorithmen eleganter und schöner als ihre iterative Form. Algorithmen zur Fakultätsberechnung und zum Quicksort aus [G95], die in Gofer notiert sind, stellen Beispiele dafür dar.
An „innerinformatischen“ Modellen betrachte ich die Turing-Maschine – als Explikation des Berechenbarkeitsbegriffes – als schöner als z.B. Markow-Algorithmen, obwohl letztere als Symbolmanipulationsvorschriften der Informatik-Vorgehensweise adäquater sind. Meine Vorliebe für die Turing-Maschine wird nicht zuletzt beeinflusst durch die überzeugende anthropomorphe Motivation, die Turing [T36] bot.
Modelle realer Systeme haben es im Allgemeinen schwerer, ästhetischen Ansprüchen zu genügen, müssen sie doch, wenn sie nicht ganz grob sind, die üblicherweise komplexe Realität widerspiegeln. Dies gilt sogar für noch relativ abstrakte Modelle wie z.B. die Typen von Chomsky-Grammatiken.
Bildliche Repräsentationen von Computerbausteinen, seien es graphische von Schaltungen oder photographische von Chips offenbaren ihren ästhetischen Reiz auf den ersten

---

[21] „Die Schönheit ist, so lautete die eine der antiken Definitionen, die richtige Übereinstimmung der Teile miteinander und mit dem Ganzen.“ [H71]

[22] Dass dies nicht generell so gesehen wird, kommt in folgendem Zitat zum Ausdruck [B96]: „In der Mathematik scheint es so zu sein, dass Schönheit eng mit Einfachheit zusammenhängt. Manche gehen so weit, diese beiden Begriffe zu identifizieren: Schönheit = Einfachheit.
Auch hier beobachtete Penrose genauer. Seiner Meinung nach ist in der Mathematik nicht Einfachheit als solche schön, sondern vor allem *unerwartete* Einfachheit.“

[23] In der Mathematik wird die Bezugnahme auf “Schönheit” nicht gescheut, wie aus zahlreichen Zitaten im Internet hervorgeht oder in Buchtiteln [H99] oder explizit zum Ausdruck kommt. Als Beispiel für letzteres ein Zitat aus [B96]: „Der berühmte englische Mathematiker G.H. Hardy (1877-1947) vertrat prononciert die Meinung, dass die Schönheit der eigentliche Maßstab für die Mathematik sei. [...] schreibt er: ‚Ein Mathematiker schafft, ähnlich wie ein Maler oder ein Dichter, Strukturen [...] Die Strukturen, die ein Mathematiker schafft, müssen so wie die der Maler und Dichter schön sein ... Schönheit ist der erste Test: Für hässliche Mathematik gibt es keinen dauerhaften Platz auf der Welt.’“
Für die Informatik war die „Ausbeute“ wesentlich karger – und eigentlich nur in einer Werbeaussage war ein entsprechendes (Selbst-)Bewusstsein vorhanden [S02]: „Software von SOLYP funktioniert nicht nur technisch perfekt, sie ist auch schön [...]“

[24] „In fact, the ecology of programming is such that overall, programmers spend over 80 percent of their time modifying code, not writing it”, said Selfridge in 1995.[DG99]

[25] By almost any standard, the software industry qualifies as a Darwinian process – from the generation of software (you combine existing code segments, then execute and weed out the bugs) to the management of the industry as a whole: eat, be eaten, or merge. [DG99]

Blick: Strukturen wie z.B. Speicherchips haben wegen ihres homogenen, modularen Aufbaus ihre eigene Schönheit, die auch zu tun hat mit ihrer Durchschaubarkeit.

Statt einer Zusammenfassung

"Ein Handwerker kann und will jedoch seine Tätigkeiten nicht wie der Spezialist in der Industrie in Funktionen aufteilen. Er muss vielen in gleicher Weise gerecht werden: den stofflichen wie den technischen, den kaufmännischen wie den organisatorischen." [S77]
Dies könnte und sollte -entsprechend modifiziert- auch als (Zusatz-)Tätigkeitsbeschreibung für Informatikerinnen und Informatiker gelten.
Wenn Francis Bacon empfiehlt, dass Wissenschaftler die Methoden des Handwerks studieren sollten und Handwerker mehr von der Wissenschaft lernen sollten, so kann dies dahingehend paraphrasiert werden, dass Informatiker jeweils die handwerklichen, technischen und wissenschaftlichen Seiten ihres Faches im Auge behalten sollten.[26]
Und auch im gymnasialen Unterricht stellt eine nicht einseitige Auffassung eine Herausforderung dar, vor allem als mit diesem Fach wohl zum ersten Mal ingenieurwissenschaftliche Problematiken angesprochen werden. [27]

Literatur

[B00] Bauer, F.L.:A computer pioneers talk: Pioneering work in software during the fifties in Central Europe,In: [I00], 11-20.

[B96] Beutelsbacher, A.: „In Mathe war ich immer schlecht...“, Vieweg, Braunschweig, 1996.

[Br96] Brooks, F.P.: The computer scientist as toolsmith II, Communications of the ACM 39, 1996, 61-68.

[BL55] o. V.: Beckmanns Neues Weltlexikon, Freytag, München, 1955.

[D75] Daumas, M.: Technikgeschichte: ihr Gegenstand, ihre Grenzen, ihre Methoden, In: Hausen, K., Rürup, R. (Hrsg.) [HR75], 31-45.

[De01] Desel, J. (Hrsg.): Das ist Informatik, Springer, Berlin, 2001.

[26] In [MP01] wird das Zusammenwirken im Titel genannten Attribute der Informatik als ein Kennzeichen der „modernen“ Wissenschaft gesehen: „Der englische Physiker Robert Boyle [...] übertrug den von Bacon theoretisch entwickelten Empirismus in die Praxis eines systematischen Experimentierens. Mit der empirischen Wende der Wissenschaft intensivierten sich die Kontakte zwischen Handwerker-Ingenieuren und theoretischen Wissenschaftlern. Nach und nach wurde so nicht nur die soziale Grenze zwischen Wissenschaftlern und technischen Praktikern durchlässig; auch ihre Denkweisen begannen sich gegenseitig zu durchdringen. Durch diesen Brückenschlag entstand auf der einen Seite die moderne Naturwissenschaft, auf der anderen Seite die schrittweise Verwissenschaftlichung der Technikentwicklung.“

[27] Damit soll nicht eine Einordnung der Informatik in den Bereich der Ingenieurwissenschaften impliziert werden. Näher wird auf diese Frage z.B. in [V102] eingegangen.

[Du01] Duddeck, H.: „Aus Schaden wird man klug …“? Wie Technik Wissen gewinnt Abhandlungen der Braunschweigischen Wissenschaftlichen Gesellschaft, **L** (2001), Braunschweig, 2001, 135-159.

[DF00] Dyson, F.J. : Die Sonne, das Genom und das Internet, Fischer, Frankfurt, 2000.

[DG99] Dyson, G.Darwin Among the Machines, Penguin Books, London, 1999.

[EB99] Encyclopaedia Britannica, Encyclopaedia Britannica Inc., Chicago, 1999.

[F94] Forman, P.: Weimarer Kultur, Kausalität und Quantentheorie 1918 – 1927, In: [vM94], 61-179.

[F97] Freksa, C., Jantzen, M., Valk, R. (Eds.): Foundations of Computer Science – Potential – Theory – Cognition, Springer, 1997.

[G98] Gal-Ezer, J., Harel, D.: What (else) should CS educators know? Communications of the ACM 41, 1998, 77-84.

[Go98] Goethe, J.W. von: Werke, Kommentare und Register, Hamburger Ausgabe in 14 Bänden, dtv, 1998.

[G95] Goos, G.: Vorlesungen über Informatik, Band 1, Springer, Berlin, 1995.

[Gr81] Gries, D.: The Science of Programming, Springer, Berlin, 1981.

[G97] Gruska, J., Vollmar, R.: Towards adjusting informatics education to information, In: [F97], 49-67.

[H95] Hartmanis, J.: Turing Award Lecture: On computational complexity and the nature of Computer Science, ACM Computing Surveys, 27, 1995, 7-16.

[H92] Hartmanis, J., Lin, H. (Eds.): Computing the Future, National Academy Press, Washington, D.C., 1992.

[H02] Hartmann, W., Nievergelt, J.: Informatik und Bildung zwischen Wandel und Beständigkeit, Informatik-Spektrum 16, 2002, 465-476.

[HR75] Hausen, K., Rürup, R. (Hrsg.): Moderne Technikgeschichte, Kiepenheuer&Witsch, Köln, 1975.

[H71] Heisenberg, W.: Die Bedeutung des Schönen in der exakten Naturwissenschaft, In: Bayerische Akademie der Schönen Künste (Hrsg.), Ensemble 2, Oldenbourg, 1971, 228-243.

[H99] Hoffmann, P.: Der Mann, der die Zahlen liebte – Die erstaunliche Geschichte des Paul Erdös und die Suche nach der Schönheit in der Mathematik, Ullstein, Berlin, 1999.

[H01] Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Boston, 2nd ed., 2001.

[I00] ICHC 2000: Mapping the History of Computing – Software Issues, Heinz Nixdorf MuseumsForum Paderborn, April 5-8, 2000.

[J77] Jagger, C.: Wunderwerk Uhr, Zollikon, 1977.

[K01] Kowar, H.: Die Wiener Flötenuhr, Technisches Museum Wien, 2001.

[K76] Knuth, D.: The Art of Computer Programming, Addison-Wesley, Boston, 1976.

[M01] Markl, H.: Das Genom, nicht der Mensch wurde entziffert, Humboldt Kosmos 77, 2001, 18-19.

[MP01] Max-Planck-Gesellschaft: Verantwortliches Handeln in der Wissenschaft: Analysen und Empfehlungen, Max-Planck-Forum 3, München, 2001.

[M719] Meyers Encyklopädisches Lexikon, Bibliographisches Institut, Mannheim, 9. Aufl.,1971-1979.

[O97] Oosterlee, K., Schüller, A., Trottenberg, U.: Durchbruch im Wissenschaftlichen Rechnen durch adaptive Mehrgitterverfahren auf Parallelrechnern, Der GMD-Spiegel 3, 1997, 15 - 18.

[R93] Rafeiner, O. (Hrsg.): Patente, Marken, Muster, Märkte – Der gewerbliche Rechtsschutz international, Manz'sche Verlags- und Universitätsbuchhandlung, Wien, 1993.

[S81] Salomaa, A.: Jewels of Formal Language Theory, Computer Science Press, Rockville, 1981.

[S03] Sasaki, T.: The political implications of science and technology, Vortrag auf der JSPS-Konferenz "Science and Society", Würzburg, 10.5.2003.

[S77] Sinz, H.: Das Handwerk, Econ, Düsseldorf, 1977.

[S02] SOLYP Informatik GmbH, http://www.solyp.de/2960.html, 2002.

[S01] Stiege, G.: Virtualität in der Informatik, Carolo-Wilhelmina XXXVI, 2001, 10-13.

[T01] Tauber, D.: Software-Entwicklung im industriellen Maßstab, In: [De01], 85-98.

[T36] Turing, A.: On computable numbers with an application to the Entscheidungsproblem. Proc. London Math. Soc. (2), 42, 230-265, 1936-7. Correction, *ibidem*, 43, 544-546, 1937.

[V102] Vollmar, R.: Von Zielen und Grenzen der Informatik, Abhandlungen der Braunschweigischen Wissenschaftlichen Gesellschaft, LI (2002), 9-24.

[V202] Vollmar, R.: Seit wann gibt es Informatik? Abhandlungen der Braunschweigischen Wissenschaftlichen Gesellschaft, LI (2002), 25-47.

[V03] Vollmar, R.: Informatik als Handwerk, Technik, Wissenschaft, Technischer Bericht 2003-10, Fakultät für Informatik, Universität Karlsruhe, 2003.

[vM94] von Meyenn, K. (Hrsg.): Quantenmechanik und Weimarer Republik, Vieweg, Braunschweig, 1994.

[vN63] von Neumann, J.: Collected Works (ed. by A.H. Taub), Pergamon Press, Oxford, 1963.

[W83] von Weizsäcker, C. F.: Wahrnehmung der Neuzeit, Hanser, 4. Aufl., 1983.

[W832] von Weizsäcker, C. F.: Heisenberg und Heidegger über das Schöne und die Kunst, In: [W83], 147-170.

[Z93] Zemanek, H.: Das Mailüfterl – ein österreichischer Aufbruch ins Computerzeitalter, In: [R93], 142-160.