Anwendungsbereiche lernen voneinander ... und woraus lernen wir?

Karl-Heinz Rödiger
Universität Bremen

Design is fun; work is not
(Harold Thimbleby 1991, p. 141)

Zusammenfassung

Ausgehend vom Motto dieser Tagung, das Lernen nahelegt, wird an vier Beispielen deutlich gemacht, daß die Bereitschaft zum wissenschaftlichen Diskurs in der Software-Ergonomie nicht gerade ausgeprägt ist. Bei den Beispielen handelt es sich um Publikationen aus den letzten beiden Jahren, in denen teilweise fundamentale Kritik am Zustand der Disziplin geübt wird. Sie werden gemeinsam mit einer richtungweisenden Arbeit, die ebenfalls die bisherige Arbeit für die Zukunft infragestellt, zusammenfassend präsentiert und in ihrer Bedeutung für das Fachgebiet diskutiert. Die Stille um diese Beiträge legt die Frage nahe, woraus wir lernen.

Im zweiten Teil des Beitrags wird über zwei Untersuchungen berichtet, die darüber Aufschluß geben, wo die Software-Ergonomie mit ihren Gestaltungsanstrengungen zwischen styleguide- und partizipationsorientierter Entwicklung steht. Hieran knüpfen sich einige Überlegungen zum Gestaltungsbegriff in der Informatik an.

1 Woraus lernen wir?

»Anwendungsbereiche lernen voneinander» heißt das von mir mitzuverantwortende, nichtsdestotrotz matte Motto unserer diesjährigen Tagung. Matt nenne ich das Motto, weil es ein wenig halbherzig zum Lernen auffordert, die lernenden Personen außen vorläßt und nicht angibt, wie dieser Lernprozeß vonstatten gehen soll. Dabei ist, wie im Beitrag gezeigt wird, wahrlich schon Gelegenheit genug zum Lernen. Insgesamt hebt das Motto auf eine Anstrengung ab, die aus meiner Sicht dem Wissenschaftsbetrieb schon seit längerem abhanden gekommen ist. Schreiben statt Lesen, Dozieren statt Lernen, die Lösung bereithalten statt nach dem Weg und den Zielen zu fragen, scheinen mir die Grundhaltungen der heutigen Umtriebigkeit zu sein.

Einmal in die Welt entlassen, läßt sich ein solches Motto nur noch schwer zurückholen. Die Anwendungsbereiche sollen voneinander lernen, aber woraus lernen wir? Schon die Einladung zur Tagung offenbart, daß wir nicht viel gelernt haben. Da wird immer noch das Residuum der Informatik beschworen, das sich ausschließlich dem Rechner, seiner Hard- und Software-Entwicklung widmen darf. Wer eine Dichotomie zwischen den »menschenorientierten Disziplinen« und den »computerorientierten Disziplinen der Informatik« konstruiert, glaubt offensichtlich immer noch, daß es diese von jeglicher Anwendung und Benutzung freien Räume informatischen Tuns gäbe. Diese Haltung erinnert mich fatal an Peter Dennings task force report »Computing as a Discipline« (Denning, P.J. et al. 1989), in dem festgestellt wird, daß »the fundamental question underlying all of computing is, "What can be (efficiently) automated?"«. Die Auswertung berechenbarer Funktionen ist die Aufgabe von Rechnern; die von Informatikern beschränkt sich nach dieser Auffassung darauf, solche Funktionen zu konstruieren.

Es mag an den Bedingungen und der Verfaßtheit des heutigen Wissenschaftsbetriebs liegen, daß Beiträge zur Eröffnung einer Tagung nicht zu einer kritischen Bestandsaufnahme genutzt werden. Dabei gäben die Befunde über den Zustand und den Erfolg der Disziplin Software-Ergonomie reichlichen Anlaß für tiefe Nachdenklichkeit. Nicht wenige Kritiker führen an, daß es in unserem Gegenstandsbereich nur zwei wirkliche Innovationen gegeben hätte, und beide seien ohne Zutun der Software-Ergonomie-Forscher zustandegekommen: 1981 mit dem Xerox Star und 1984 mit dem Macintosh von Apple. Auch die Style Guides als Marksteine ergonomischen Gestaltungswillens sind Produkte von Anbietern, die sie eher mit Blick auf Marktanteile geschaffen haben, denn mit der Absicht, wissenschaftliche Erkenntnisse umzusetzen. Alles andere im Fach, so die Kritiker, sei Nachlaufforschung, die nun, ex post, begründen könne, warum z.B. Direkte Manipulation so erfolgreich ist.

Ich will mich hier nicht mit dieser Rundum-Kritik an unseren Forschungsbemühungen beschäftigen. Eine Auseinandersetzung damit gerät - führt man sie nicht präzise und detailliert, wozu an dieser Stelle der Raum fehlt - sehr schnell in den Bereich der Behauptungen und Rechthabereien. Ich gestehe jedoch, daß ich diese Position verstehe. Mehr als solcherart grundsätzlichen Infragestellens der Software-Ergonomie berühren mich vier differenziertere Kritiken an unserem Lehr- und Forschungsgebiet aus jüngerer Zeit; bei dreien irritiert mich zusätzlich die Resonanz, die diese Kritiken in unseren Reihen ausgelöst hat: nahezu keine. Bei der vierten gab es bisher keine Chance zur Stellungnahme: sie ist gerade erst erschienen, war jedoch für jeden, der in diesem Gebiet forscht, abzusehen.

2 Fehlende Wirkung

1992 schreibt Peter Gorny in einer »Zwischenbilanz Menschengerechte Gestaltung von Software«: "Software-ergonomische Methoden und Werkzeuge - z.B. softwareergonomische Analysen (Aufgaben- und Benutzeranalysen), Gestaltungsansätze (Dialoggestaltung und Informationspräsentation) und Evaluationskomponenten (Evaluation der Benutzbarkeit) - finden nur als wissenschaftliche Fragestellungen Einzug in AuT-Projekte (Arbeit und Technik, der Verf.). In der normalen Softwareentwicklungspraxis wird auf ihre Anwendung verzichtet" (Gorny 1992, S. 13). Walter Volpert hat diese Kritik in seinem Beitrag zur Software-Ergonomie-Tagung 1993 aufgegriffen und einen »merkwürdigen Widerspruch « konstatiert: »Die Bildschirm-Oberflächen werden hübscher, aber die Arbeitsplätze werden nicht besser« (Volpert 1993, S. 53).

Die deutschsprachige Software-Ergonomie hat ihr eigenes Profil in einem ganzheitlichen Gestaltungsansatz gefunden. In jedem grundlegenden Beitrag, in dem das Gebiet ausdifferenziert wird, wird dieser Ansatz entweder mit dem "Zwiebelmodell" (Rödiger 1985, S. 460) oder der "modifizierten Leavitt-Raute" (Oberquelle 1991, S. 11) veranschaulicht. Dennoch scheint, wie die Zitate belegen, von diesem ganzheitlichen Ansatz in der Praxis nicht viel übrig zu bleiben. Ausgenommen natürlich in den Arbeit und Technik-geförderten Forschungsvorhaben, sie sind von der Projektbeantragung her schon einem solchermaßen ganzheitlichen Ansatz verpflichtet. Leider wird auch in diesen Projekten des öfteren diagnostiziert, daß nach Wegfall der Fördermittel auch die ganzheitlichen Gestaltungsmaßnahmen entfallen.

Man muß nicht sehr tief nachgrübeln, um hierfür Erklärungen zu finden. Wenn sich schon mit wenig Aufwand Akzeptanz bei neuen Systemen erreichen läßt, warum dann viel treiben? Für mich erstaunlicher, ja erschreckender ist die Feststellung, daß auch auf dem Gebiet der Arbeitsmittelgestaltung, der technischen Auslegung von Benutzungsschnittstellen, in der Praxis eine hohe Unsicherheit herrscht und viele Fehler gemacht werden, die wir schon ausgerottet gesehen hatten. Ich beziehe mich hierbei auf meine vielfältigen Praxiskontakte, die mir einen Einblick in die Qualität software-ergonomischer Lösungen erlauben. Es werden nach wie vor Masken entworfen, die wir schon 1983 kritisiert haben; Menüs sind unstrukturiert und verstoßen gegen Grundsätze der Objektorientierung; Fenster werden kreiert, die weder aufgabenangemessen noch Styleguide-konform sind (mit einem besonders schlechten Beispiel geht hier die Fa. Microsoft voran); und in sog. Fehlermeldungen taucht immer noch der erhellende Hinweis »Fehler« auf.

Daß die Praxis einmal wieder »nichts gelernt« hat, können wir ihr sicher vorwerfen, hilft der Reputation unserer Disziplin aber auch nicht auf die Beine. Eine zurückhaltendere Grundhaltung sollte uns dazu treiben, die Fehler bei uns zu suchen. Davon sehe ich einige:

- Transfer software-ergonomischen Wissens haben wir bisher nach der hydraulischen Theorie des Lernens (P.M. Davies 1969) betrieben: Wissen ist eine Flüssigkeit, die sich vornehmlich in unseren Köpfen und in Büchern befindet; und diese Flüssigkeit verspritzen wir am liebsten vor einer großen Hörerschaft. Daß sich in einem so praxisorientierten Feld, wie der Softwareentwicklung, damit nur geringe Lernerfolge erzielen lassen, sollte sich herumgesprochen haben.

- Auch wenn ich gerade Bücher der hydraulischen Theorie zugeordnet habe: Das software-ergonomische Lehrbuch, das das Feld in toto darstellt und dazu noch Handlungsanleitung bietet, ist noch immer nicht geschrieben. Das praxisorientierte deutschsprachige Pendant zu Tog on Interface (Tognazzini 1992) fehlt ebenfalls.

- Transfer läßt sich auch nicht allein über Normen und Standards, die wir zur Zeit reichlich - so innerhalb der ISO 9241 allein 17 Teile - produzieren, erzielen. Sie sind wegen ihrer zeitlosen Allgemeinheit eher zur Abschreckung der Praxis geeignet.

- Wenn wir über software-ergonomische Gestaltung reden, zeigen wir schlechte Beispiele und hoffen, daß unsere Hörer aus der Kritik an ihnen die gute Gestalt herausfinden. Wir befinden uns hier in einem grundsätzlichen Dilemma, das Christopher Alexander auch in anderen Gestaltungsdisziplinen konstatiert hat: »... the procedure suggests no direct practical way of identifying good fit. We recognize bad fit whenever we see a high spot marked by ink. But in practice we see good fit only from a negative point of view, ... (Alexander 1964, p. 22).

Wie aber macht man's dann? Die Patentlösung muß auch ich schuldig bleiben; die besten Erfahrungen habe ich gemacht, wenn wir uns auf Entwicklung eingelassen haben. Wir haben für die Praxis selbst entwickelt und versucht, gute Beispiele zu geben. Außerdem haben wir uns beratend an Entwicklungsprozessen beteiligt, gemeinsam gute Schnittstellen entwickelt und diese dann verbessert.

Peter Gorny hat mit seiner Bilanz an den Grundfesten unseres Tuns, dem Streben nach auch praktischer Wirksamkeit gerüttelt. Wo bleibt der Aufschrei, der Protest? Was mir in unserem Arbeitsgebiet fehlt, sind nicht die schnellen Lösungen zu

diesem Problem, die Transfer-Projekte, die von diesem Dilemma leben, sondern der Diskurs zu diesem Vorwurf.

3 Ungesicherte Forschungsergebnisse

1993 schreiben Nachreiner & Mesenholl in einer »Bilanzierung vorliegender Forschungsergebnisse zur Arbeit an Bildschirmgeräten«: Eine methodische Überprüfung der Ergebnisse zeige als erstes Defizit den »insgesamt eher unbefriedigenden Stand der Forschungen zur Software-Ergonomie - von der Gegenstandsbestimmung über die Definition der Konzepte, die berücksichtigten Kriterien, die angewandten Methoden und die untersuchten Fragestellungen« (Nachreiner & Mesenholl 1993, S. 604). Und sie kommen zu dem Schluß, daß die Software-Ergonomie »das Bild einer wenig strukturierten Disziplin« bietet (a.a.O., S. 602). Vornehm hält sich Friedhelm Nachreiner auch in einem weiteren Beitrag zurück, »einzelne Arbeiten, Personen oder Forschungsgruppen zu kritisieren oder an den Pranger zu stellen« (Nachreiner 1994, S. 53). Vielmehr geht es ihm darum, »eine sich etablierende Disziplin auf ihre nachlässige und z.T. nicht verantwortbare Behandlung ihrer Methodenprobleme hinzuweisen« (l.c.).

Es mag wie "Nachklappen" klingen, aber Nachreiners Original-Bilanz an den Projektträger Arbeit und Technik liegt mir nicht vor. 1990 habe ich mehrere Vorträge über Objektorientierung gehalten und mir dazu auch einschlägige Arbeiten zu objektorientierten Benutzungsoberflächen angeschaut, um herauszufinden, ob es Indizien für eine mögliche Überlegenheit dieses Prinzips gegenüber Funktionsorientierung im Bereich der Schnittstellengestaltung gibt. Was ich fand, war, daß in der mir vorliegenden Literatur Objektorientierung mit Direkter Manipulation oder mit graphischen Oberflächen gleichgesetzt wurde, allesamt aber das Hohelied auf die sog. neuen Interaktionstechniken sangen. Ich habe mich damals nicht getraut, von Methodenproblemen der Software-Ergonomie zu sprechen. Daß es diesen Arbeiten sehr wohl an interner Validität fehlte, war mir bewußt. Vorgetragen habe ich damals, daß ich wegen dieser Probleme aus allen mir vorliegenden Arbeiten keinen Vorteil einer objektorientierten Auslegung von Benutzungsschnittstellen gegenüber funktionsorientierten feststellen könne, die Arbeiten letztlich nur einen Schluß zulassen: Wer sich in seiner Oberfläche auskennt - sei sie nun objekt- oder funktionsorientiert -, ist darin gut und allen Novizen überlegen (sic!). Die damalige Euphorie um graphische versus zeichenorientierte Oberflächen, die mit den anderen Sachverhalten immer wieder vermengt wurde, habe ich nicht verstanden.

Unter einer Nachwirkung dieser nie wissenschaftlich überprüften Euphorie leiden wir heute noch: Es ist der sinnlose Drang nach Symbolen dort, wo Zeichen angemessener wären. Wenn jemand glaubt, ein Bild sage immer mehr als tausend Worte, so möge er sich die Beispiele von Barbara Lauter (Lauter 1987) oder die Vorschläge der DIN E 40 107 Teil 1 anschauen. Um diese Symbole in der täglichen Arbeit handhaben zu können, müßte ich mir deren Bedeutung mittels Zeichen auf einem Post-it notieren und an den Bildschirm kleben.

Doch zurück zu Nachreiners Bilanzierung: Wiederum irritiert mich deren Rezeption unter den Software-Ergonomie-Forschern. Kein Aufschrei, kein Protest, vor allem aber auch kein Diskurs zum Zustand unseres Forschungsgebiets.

4 Unzureichende Lehre

In seinem Beitrag »Informatik auf der Mauer« setzt sich Jörg Pflüger mit den in jüngerer Zeit erschienenen Artikeln zur Gegenstandsbestimmung der Informatik und zur Curriculardebatte auseinander. Er folgert aus einer Analyse des Gegenstandsbereichs, »daß wir zu den ingenieurwissenschaftlichen und mathematischen Säulen als drittes Standbein eine geistes- und sozialwissenschaftliche Fundierung in die Informatikausbildung aufnehmen müssen« (Pflüger 1994, S. 255). Man könnte ob dieser Feststellung eines theoretischen Informatikers frohlocken, würde er nicht folgendermaßen fortfahren: »Es kann ... nicht genügen, den anderen Fächern ein weiteres hinzuzugesellen, sondern es muß darum gehen, die Spanne zwischen Verstehen, Formalismus und Technik als solche zu lehren« (l.c.). Und mit einem Seitenhieb auf die Angewandte Informatik in Bremen: »Allzuoft wird dabei nur der kleinste gemeinsame Nenner einer bastelnden Unverbindlichkeit ausgebildet« (a.a.O, S. 256). Nicht nur die Bremer Lehrenden haben bisher angenommen, daß sie den zuvor formulierten Ansprüchen halbwegs gerecht würden.

Bis hierhin kann man Pflügers Kritik noch nachvollziehen und mit ihr in eine neue Curriculardebatte insbesondere auch um die Ausbildung in Software-Ergonomie eintreten. Zu einer solchen Debatte fordert auch Horst Oberquelle auf, der in einem nachdenklichen Beitrag nach neuen Inhalten und Formen der Ausbildung in Software-Ergonomie sucht (Oberquelle 1994). Mit dem, was Oberquelle unter Orientierungswissen behandelt und vermitteln will, kommte er Pflügers Ansprüchen an eine dritte Säule nahe: »Das Orientierungswissen umfaßt alle die Teile, die mit Menschenbild, Wertmaßstäben und Bewertungen zu tun haben« (a.a.O., S. 25).

Mit der Feststellung »Im Hinblick auf die heutige technikorientierte Ausbildung müßte also vordringlich eine andere "Sprache" vermittelt werden, in der Differenzen und Konflikte verstehbar sind und Irrtümer ausgedrückt werden können« (Pflüger a.a.O., S. 256, Hervorhebung im Original) kommt der Autor zu dem überraschenden Schluß: »Es erschiene mir nicht abwegig, wenn Informatiker lernen würden, Gedichte zu interpretieren und historische Prozesse zu beurteilen« (l.c.).

Mit dieser Forderung nach der Einübung des »hermeneutischen Blicks« geraten wir an die Grenzen dessen, was im Rahmen der üblichen Hochschullehre zu leisten ist. Ich interpretiere Pflüger nicht so eng, daß ich Friedrich Schillers Balladen oder Ernst Jandls Lautgedichte zum Gegenstand von Übungen und Leistungsnachweisen machen müßte. Sein Anliegen des Sicheinlassens auf das Differente, des Verstehenwollens anderer Sprachwelten, des Umgehens mit Mehrdeutigkeiten läßt sich meines Erachtens auch in einem praktischen Softwareentwicklungsprojekt für einen den Studierenden fremden Gegenstandsbereich einlösen. Allerdings bedarf es hierzu geraumer Zeit, die dieses Sicheinlassen ermöglicht.

In dieser Beziehung habe ich mit dem Bremer Projektstudium über eine Laufzeit von vier Semestern gute Erfahrungen gemacht. In meinem gerade laufenden Projekt "CAD im Schneiderhandwerk" lernen die Studierenden die Probleme und die Sprache des Handwerks; sie müssen damit umgehen, daß in den Betrieben Schnitte nach unterschiedlichen Methoden konstruiert werden; sie lernen, daß nicht alles, was technisch machbar ist, auch gewollt wird. Sie haben die dramatischen Veränderungen in der Textilbranche kennengelernt. Sie lernen zur Zeit, daß "easy-to-use"-Oberflächen in Handwerksbetrieben eine andere Bedeutung haben als an der Hochschule. Und sie haben vor allem eines schon gelernt: Daß sich die Arbeitsweisen von (zukünftigen) Benutzern nicht mit den Moden der Softwareentwicklung verändern. Unzweifelhaft ist Objektorientierung, wenn sie von den entsprechenden Sprachmitteln gestützt wird, ein gutes Programmierprinzip, das auch dem Stand der Kunst entspricht. Unzweifelhaft ist objektorientierte Abstraktion ein gutes Modellierungsprinzip im Entwurfsprozeß; sicherlich kann ich auch mit den entsprechenden Werkzeugen eine gute Systemspezifikation entwickeln. Daraus aber nun den Schluß zu ziehen, Benutzer, gleich welcher Anwendungsdomäne, hätten immer schon objektorientiert gearbeitet, die ganze Welt bestünde nur aus Objekten oder Objektorientierung sei ein "natürliches Prinzip", ist mehr als keck. Wer sich mit dem unverstellten Blick auf die Praxis einläßt, wird hierfür keinen Beleg finden.

Nicht überall in den Informatik-Studiengängen hat man für dieses Sicheinlassen auf andere Erfahrungswelten vier Semester Zeit; und auch in Bremen ist nicht jedes Projekt in diesem Sinne ein Erfolg. Sprache und Probleme von Anwendungsberei-

chen lernt man nicht über Vorlesungen verstehen. Es macht einen großen Unterschied, ob ich von den Problemen des Handwerks erzähle oder ob die Studierenden in die Betriebe gehen können und ihre Erfahrungen durch teilnehmende Beobachtung erwerben. Hierzu aber benötigt man ein "Feld", das diese zusätzlichen Belastungen erträgt, besser noch unterstützt.

Vor der anderen Forderung Pflügers, »daß die Vermittlung der drei Sichtweisen (mathematisch, ingenieur- und sozialwissenschaftlich, der Verf.) wesentlich in einer Konfrontation ihrer Methoden besteht« (Pflüger 1994, S. 256) muß ich als jemand, der ebenso wie Oberquelle Theorien und Methoden der Sozialwissenschaften »autodidaktisch« erworben hat und vermutlich »mehr schlecht als recht« vermittelt, passen. Da ich den »selektiven Umgang mit den Nachbardisziplinen« (Kubicek 1984) ablehne, der mich möglicherweise dazu treibt, nur das zu lehren, was ich verstanden habe oder was in meine Denkschemata paßt, darf ich eine Veranstaltung dieses Inhalts nur zusammen mit den Kollegen der entsprechenden Disziplinen (Arbeitswissenschaft, Psychologie) anbieten.

5 Unzeitgemäße Forschungsgebiete

In einer kleinen Nebenbemerkung seines schon zitierten Aufsatzes bemerkt Horst Oberquelle, ein besonderes Problem ergäbe sich aus der Tatsache, »daß die heute ausgebildeten Software-Ergonomie-Experten in ihrer beruflichen Praxis kaum noch solche Systeme bauen werden, die wir heute einigermaßen verstanden haben (z.B. GUIs), die genormt werden und die wir in unseren Lehrveranstaltungen behandeln« (Oberquelle 1994, S. 24). Er hebt damit auf einen Beitrag Marc Weisers zur zukünftigen Arbeit mit und an Informatiksystemen ab.

Aus einer Kritik am heutigen PC-Einsatz formuliert Weiser seine Perspektive "allgegenwärtiger Computernutzung": »... rather than being a tool through which we work, and thus disappearing from our awareness, the computer too often remains the focus of attention« (Weiser 1993, p. 76). Und weiter: »Getting the computer out of the way is not easy. This is not a graphical user interface (GUI) problem, but is a property of the whole context of usage of the machine and the attributes of its physical properties ... The problem is not one of "interface"« (l.c.). Damit wird nicht nur die Adäquatheit unserer Lehre in Software-Ergonomie infrage gestellt; damit stehen unsere Forschungsanstrengungen der vergangenen Jahre, die in der Hauptsache um den PC, um graphische Benutzungsschnittstellen oder die Arbeit mit PCs im allgemeinen kreisten, zur Disposition.

Es wäre nicht das erste Mal, daß aus dem Xerox Palo Alto Research Center (PARC) nicht nur Ideen für die zukünftige Arbeit, sondern ein hard- und softwaremäßig realisiertes System käme, daß unsere Arbeits- und Lebenswelt für die kommenden Jahre bestimmen sollte. Man erinnere sich in diesem Zusammenhang: Im April 1981 kam das Xerox 8010 Star Information System auf den Markt, das in Architektur und Handhabung viele völlig neue Eigenschaften aufwies, die uns heute so vertraut sind: Client-Server-Architektur, Lokales Netz, graphikfähiger Bildschirm, Maus, Objektorientierung, Symbole statt Zeichen zur Kennzeichnung von Objekten, Direkte Manipulation etc. Die von D.C. Smith et al., Mitarbeitern des Star-Projekts bei Xerox, vorgelegten Entwurfsprinzipien, stehen so oder sehr ähnlich heute in jedem Styleguide für graphische Benutzungsoberflächen: »Familiar user's conceptual model, seeing and pointing versus remembering and typing, what you see is what you get, universal commands, consistency, simplicity, modeless interaction, user tailorability« (Smith et al. 1982, p. 248).

Den Arbeiten aus dem PARC mag man noch entgegenhalten, daß sie zur Zeit noch mehr von Visionen getragen ist, deren "Beforschen" wegen der uns größtenteils fehlenden, bei Xerox maßgeschneidert entwickelten Hardware schwer fällt. Auch will ich nicht verkennen, daß mit den "aktiven Kennzeichen" (active badges) und dem Beispiel der Verortung von Personen tiefe Eingriffe in die Persönlichkeitsrechte verbunden sind, die ich mit meiner Auffassung von verantwortlichem Handeln in der Informatik nicht vereinen kann (Capurro et al. 1993). Dennoch sollte uns schon der "Abgesang" auf unsere zentralen Forschungsgegenstände dazu bringen, uns mit diesen Perspektiven verstärkt zu beschäftigen.

Grundsätzlicher sollte uns eine Kritik treffen, die Herbert Kubicek und Wolfgang Taube im Dezember 1994 unter dem Titel »Die gelegentlichen Nutzer als Herausforderung für die Systementwicklung« im Informatik-Spektrum publiziert haben. Ausgehend von der Annahme, daß arbeitsorientierte Informatiksysteme »vermutlich sogar einen relativ abnehmenden Anteil am Gesamteinsatz der Informationstechnik, zumindest wenn man den gängigen Arbeitsbegriff unterstellt und Computerspiele oder Bürgerinformationssysteme nicht ohne weiteres darunter subsumiert« (Kubicek & Taube 1994) kritisieren sie die grundsätzliche Ausrichtung der Software-Ergonomie: Sie habe als zentrales Paradigma »Softwaregestaltung ist Arbeitsgestaltung«, von daher einen Begriff von Benutzern, der mit denen alltagsorientierter Systeme wenig gemein hat, und würde deshalb für diese Systeme keine Konzepte bereithalten.

An zwei Beispielen (Electronic Cash und elektronischer Bibliothekskatalog) demonstrieren die Autoren zwei typische Verhaltensweisen, mit denen Software-Entwickler auf die mangelnde Kenntnis über den Gegenstandsbereich und dessen

Nutzern reagieren: "Der Schluß von sich auf andere" und "Der Schluß von einer Gruppe auf eine andere". Beide Projekte entwickelten sich als Fehlschläge. Aus einer Analyse von Arbeiten zur Benutzerklassifikation kommen die Autoren zu dem Schluß, daß eine solche Typisierung nicht weiterhilft, da sie einerseits statisch ist, sich damit andererseits Benutzer nur schwer definieren lassen, die neben ihrem professionellen Umgang mit einem arbeitsorientierten Informatiksystem an alltagsorientierten Systemen beispielsweise als gelegentliche Benutzer auftreten.

Kubicek & Taube plädieren für eine situative Sicht auf Nutzung, in der uns die aus der Leavitt-Raute (Oberquelle 1991) bekannten Einflußgrößen auf Technikentwicklung Mensch, Arbeit und Organisation in verallgemeinerter Form als Persönlichkeit, Anwendungsinhalt und Milieu entgegentreten. In einer solchen Sicht treten Arbeit und Arbeitende als eine Spezialisierung auf. Die Autoren haben keine neuen Gestaltungsdimensionen erfunden; sie haben uns darauf hingewiesen, daß sich unser Forschungsfeld öffnen muß, wenn wir den gewandelten Formen der Rechnernutzung gerecht werden wollen. Außerdem haben sie mit dem Vorschlag, die Nutzungssituation, beschrieben durch das Tripel Persönlichkeit, Inhalt und Milieu, in den Fokus zu nehmen, einen Weg zu benutzungsangemessener Forschung und Entwicklung gewiesen.

In der Tat haben wir uns in der Vergangenheit im wesentlichen auf Arbeit und arbeitsorientierte Informatiksysteme, oft sogar mit der weiteren Einschränkung Büro, beschränkt. Wir haben in der Software-Ergonomie-Forschung den Wandel in der Nutzung des Computers vom Werkzeug im Arbeitsleben zum Medium in allen gesellschaftlichen Bereichen zu wenig zur Kenntnis genommen (Coy 1994). Einen Großteil zukünftiger Computernutzung wird nicht das Rechnen, das computing, nicht die abhängige Erwerbsarbeit in Organisationen einnehmen, sondern die mediale Nutzung von Dienstleistungsangeboten in den sog. Infobahnen. Schon heute haben sich Standards in der Verwendung des Internets oder von World Wide Web (WWW) anarchisch und neben den bekannten software-ergonomischen Gestaltungsgrundsätzen herausgebildet.

Unser Nutzungsverhalten hat sich durch Rechnernetze schon verändert. Es wird sich durch den Ausbau dieser Netze weitergehend verändern; z.B. bezogen auf unsere Freizeitgestaltung nachhaltig dadurch, daß uns immer mehr Dienstleistungsunternehmen ihre Arbeit machen lassen, seien es nun Banken, Versicherer, Reiseveranstalter oder Transportunternehmen. Für diese Anwendungen, mit denen wir uns nun selbst herumschlagen dürfen, haben wir in der Vergangenheit - siehe Kubicek und Taube - zu wenig getan.

6 Wo bleibt das Positive?

Ich höre sie schon, die Rufe nicht nur der Veranstalter nach der positiven Wende im Vortrag, die doch noch Anlaß zu Ermutigung und Aufbruchstimmung bietet. Ich will sie nicht schuldig bleiben, allein schon, um zu demonstrieren, daß wir nicht nur kritisieren, sondern gelegentlich auch entwickeln. Es gilt, von zwei Untersuchungen zu berichten, die unter einem bestimmten Blickwinkel Antwort auf die Frage nach einer Vorgehensweise bei der software-ergonomischen Gestaltung geben sollten.

Ziel der Untersuchungen war herauszufinden, wie brauchbar die herstellereigenen Styleguides einschließlich der sie unterstützenden Werkzeuge bei der Entwicklung interaktiver Software sind. Als Derivat dieser Fragestellung sollte eine Vorgehensweise zwischen den Polen styleguide-, gleichbedeutend mit ausschließlich entwicklerbezogener, und partizipationsorientierter Entwicklung erarbeitet werden. Diesen Fragen wurde in zwei Diplomarbeiten nachgegangen.

Hierzu wurde in der ersten Arbeit (Rosebrock & Siegler 1993) zunächst in enger Kooperation mit Angestellten einer Bank ein Prototyp für den Bereich Kontokorrent entwickelt und evaluiert. Die Studenten brachten in diese Kooperation ihre software-ergonomischen Kenntnisse ein. Die Angestellten verfügten neben ihren Fachkenntnissen über langjährige Erfahrungen mit zeichenorientierten Bankanwendungen. Diese Fähigkeiten wurden in der Entwicklung eines graphikorientierten Prototypen zusammengeführt. Rosebrock und Siegler haben diesen Prototypen dann so genau, wie mit Hilfe des Common User Access (CUA) in der Version von 1991 (IBM 1991) und den sie unterstützenden Werkzeugen (IBM Toolkit) möglich, umgesetzt und das Ergebnis wie den Werkzeugeinsatz bewertet.
In einer zweiten Diplomarbeit hat Szczepanek (Szczepanek 1994) den gleichen, von den Bankangestellten mitentwickelten Prototypen mit Hilfe des Windows Application Design Guide, den weiteren Microsoft-Dokumenten und mit den sie unterstützenden Werkzeugen Visual C++ und Application Studio unter Windows umgesetzt. Auch er hat sowohl das Ergebnis seiner Implementierung wie den Werkzeugeinsatz bewertet.

Aus beiden Arbeiten lassen sich folgende Ergebnisse zusammenfassen:

- Der in der engen Kooperation mit den bankfachlichen Experten zustandegekommene Prototyp ist sowohl der CUA- wie der Windows-Implementierung bezüglich Aufgabenangemessenheit und Ästhetik überlegen. Allerdings hat auch die Kooperation einen grundsätzlichen Design-Fehler nicht verhindern können.

- Die CUA-Implementierung erfüllt die Anforderungen der Bankangestellten besser als die unter Windows.

- Der CUA ist als Styleguide für Entwickler hilfreicher als die entsprechenden Microsoft-Dokumente.

- Die Werkzeugunterstützung zur Entwicklung graphischer Benutzungsoberflächen von Microsoft ist besser als die von IBM. Komfortable Werkzeuge zur Entwicklung von CUA-Oberflächen werden nur von Fremdherstellern angeboten.

- Zwischen den Windows-Dokumenten Application Design Guide, Interactive Design Guide und Visual Design Guide gibt es zahlreiche Widersprüche. Immerhin aber bietet Microsoft im Gegensatz zu IBM Online-Unterstützung bei der Entwicklung an.

Zwar ist es als Nebenprodukt dieser Untersuchungen durchaus von Interesse, die Anbieter bezüglich ihrer Styleguides und der sie unterstützenden Werkzeuge miteinander zu vergleichen; im Mittelpunkt der Untersuchungen stand allerdings die Frage nach der Gebrauchstauglichkeit der Styleguides und nach einer generellen Vorgehensweise im Entwicklungsprozeß. Hierauf bezogen liegen deutliche Belege für die Vorzüge kooperationsorientierter Entwicklung vor, zeigt sich doch, daß die zwischen Bank- und Informatikexperten ausgehandelte Dialogschnittstelle den styleguide-konformen Produkten in mehreren Punkten deutlich überlegen ist. Der Begriff der Kooperationsorientierung wird hier im Sinne von Weltz & Ortmann (1992) als gemeinsamer Lern-, Entscheidungs- und Entwicklungsprozeß und in deutlicher Abgrenzung zu einem Partizipationsbegriff verstanden, der auf Manipulation und Wissensenteignung (Volpert 1992, S. 177) ausgerichtet ist.

Wenn auch Fallstudien, wie diejenigen, über die hier berichtet wird, durchaus Fragen nach der Übertragbarkeit und Generalisierbarkeit der Ergebnisse aufwerfen, so sind sie doch geeignet, zwei sehr einfach anmutende, schon länger diskutierte Thesen zu erhärten.

1 Mit Normen und Styleguides allein lassen sich allenfalls aufgabenunabhängige (Teile von) Benutzungsschnittstellen angemessen entwickeln.

2 Benutzer-Kooperation verhindert keine Design-Fehler; zur aufgabenangemessenen Gestaltung ist sie jedoch unabdingbar.

Man kann diese Thesen auch zu einer zusammenfassen: Das Problem der angemessenen Gestaltung von Benutzungsoberflächen ist trotz fünfzehn Jahren Forschung und Entwicklung zur Software-Ergonomie, trotz aller Fortschritte und trotz der bislang vorliegenden Standards ungelöst und wird vermutlich auch weiterhin ungelöst bleiben. Warum soll in der Informatik etwas zu leisten sein, was andere mit Gestaltung befaßte Disziplinen trotz wesentlich längerer Tradition nicht geschafft haben?

Stadtplaner bescheren uns Einkaufsstraßen, in den sich nach Geschäftsschluß niemand mehr aufhalten möchte; im neuen Bahnhof Kassel Wilhelmshöhe waren keine Toiletten vorgesehen; den neuerbauten Düsseldorfer Hauptbahnhof mußte man nachträglich wieder aufreißen, um Fahrstühle für Behinderte einzubauen; Bauingenieure dienten uns Spannbeton an, der nun mit hohem Aufwand saniert werden muß; alle drei großen Flugzeug-Unternehmen mußten schon Konstruktionsfehler einräumen. Man kann Gebrauchsgegenstände kaufen, die hervorragend designed aber dysfunktional sind (vice versa): Mein vorheriger Anrufbeantworter pflegte von Zeit zu Zeit Anrufe mit dem Text »Ich danke für Ihren Anruf« einzuleiten, um die Verbindung dann zu unterbrechen, ohne daß der Anrufer irgendeine Chance der Rückäußerung gehabt hätte.

Was aber begeistert uns an den Reisterassen von Banaue (Philippinen)? Es ist die grandiose Verbindung von Funktionalität und Ästhetik: Das Wasser fließt nach einem ausgeklügelten System, Reis kann auch an den steilsten Hängen angebaut werden, und die Felder in ihrer scheinbar willkürlichen Verteilung und ihren unterschiedlichen Bearbeitungs- und Reifezuständen sind ein Augenschmaus.

»Irren ist menschlich« überschreibt Donald Norman sein Kapitel über Design-Fehler und begründet, aus welchen Situationen heraus Menschen Fehler machen (Norman 1989). Da solche Situationen immer wieder eintreten können, werden auch immer wieder Fehler gemacht. Insofern relativiert sich zumindest ein Teil der immerwährenden Diskussion um die Softwarekrise (Spillner 1994), den um die mangelnde Produktqualität. Auch andere Disziplinen mit wesentlich längerer Lehr- und Forschungserfahrung kennen diese Probleme und haben sie nicht endgültig lösen können. Wir sollten uns jedoch die Erfahrungen der anderen mit Gestaltung befaßten Disziplinen zu eigen machen, daraus lernen und unsere eigenen Methoden und Techniken ständig zu verbessern.

7 Zur Gestaltungsdiskussion in der Informatik

In der Informatik wird entlang einem wie auch immer beschaffenen Vorgehensmodell ermittelt, analysiert, definiert, spezifiziert, entworfen, implementiert, getestet, eingesetzt und gewartet. All diese Tätigkeiten im Softwareentwicklungsprozeß kennen auch ein Ergebnis: Anforderungsdefinition, Systemspezifikation, Entwurf, implementiertes System, getestetes System, eingesetztes System, gewartetes System. Und irgendwo dazwischen wird offensichtlich auch gestaltet, glaubt man der Informatik-Literatur und meinem Beitrag, in dem dieser Begriff vielfältig vorkommt. Wo aber passiert das zwischen Analyse, Spezifikation, Entwurf und Implementierung? Und was ist das Ergebnis? Gestaltung offensichtlich nicht, der Begriff bezeichnet einen Prozeß, allenfalls einen Zustand. Ist es die Gestalt, die gute Gestalt, das gut gestaltete System? Was aber ist das? Wie ist es definiert? Etwa durch »Die zehn Tugenden guter Software« der Gütegemeinschaft Software (1987)?

In der Informatik gibt es seit geraumer Zeit eine Diskussion um den Gestaltungsbegriff. Schon früh hieß es »Softwaregestaltung ist Arbeitsgestaltung« (Hacker 1987), womit allerdings mehr die Wirkung von Gestaltung in der Softwareentwicklung, denn ihr Wesen beschrieben wurde. Wenn Wolfgang Coy in der Informatik eine universelle Gestaltungswissenschaft sieht, so ist diese Sicht gegen technikzentrierte Auffassungen von Informatik gerichtet, gibt jedoch ebenfalls keine Auskunft darüber, was Gestaltung sein könnte, und wo im Entwicklungsprozeß sie Platz greift (Coy 1989). Auch Arno Rolf verwendet einen breiten Gestaltungsbegriff, der Technikfolgenabschätzung mit einschließt, ohne zu explizieren, wie der in der Informatik gefüllt werden könnte und wo er in der Softwareentwicklung ansetzt (Rolf 1992). Walter Volpert tritt gegen den »gewissermaßen imperialistischen« Anspruch der Informatik an, alle mit Gestaltung befaßten Disziplinen vereinnahmen zu wollen: »Wenn sie ... alle diese Disziplinen in sich aufsaugen möchte, so muß sie dies auch substantiell, also hinsichtlich der Gesamtheit der Erkenntnisse tun. Wenn sie sich aber als einen Teil der universellen Gestaltungswissenschaft sieht, dann muß sie ihren eigenen Beitrag und Schwerpunkt definieren« (Volpert 1992, S. 173). Frieder Nake definiert Gestalten als »das Herstellen mit Ungewißheit, das unsichere Herstellen. Das Herstellen mit einer Gewißheit will ich im Gegensatz hierzu die *Konstruktion* nennen« (Nake 1992, S. 200, Hervorhebung im Original). Dirk Siefkes versteht unter Gestalten »die unruhige Einheit aus Verstehen und Herstellen« und »Gestalten ist Auf-menschliche-Weise-Machen« (Siefkes 1992, S. 112).

Diese Übersicht über die Gestaltungsdiskussion in der Informatik beansprucht keine Vollständigkeit. Sie zeigt jedoch schon, daß hier mit sehr unterschiedlichen Gestaltungsbegriffen argumentiert wird, die verschiedene Bedeutungshöfe und

Konnotationen haben. Sie sind nur schwer gegeneinander zu diskutieren und wegen ihrer unterschiedlichen Ansätze kaum für die Softwareentwicklungspraxis tauglich. Neben emphatischen Begriffen des Gestaltens (Siefkes), stehen solche die auf beabsichtigte Ziele (Rolf), (unbeabsichtigte) Wirkungen (Hacker), Werte (Nake) oder Konsequenzen (Volpert) abheben. Die Vielfalt der Bedeutungsgehalte von Technikgestaltung hat gerade Hans Dieter Hellige (Hellige 1994) in einer schönen Arbeit erhellt.

Wo aber im Software-Lebenszyklus findet Gestaltung statt? Hier liegt meines Erachtens die eigentliche Krux des Gestaltungsbegriffs in der Informatik. Gestaltung läßt sich nicht irgendwo zwischen Entwurf und Implementierung verorten und dort gegebenenfalls gezielt beeinflussen. Entgegen der reinen Lehre, nach der auch die Systemspezifikation noch frei von allen Gedanken an das WIE der technischen Realisierung sein soll, weiß jeder Entwickler, daß man sich zu keinem Zeitpunkt völlig frei von Gedanken an die (auch falsche) Realisierung machen kann. Und so entsteht dann schon bei der Analyse der frevlerische Gedanke an das Icon, das genau den Sachverhalt symbolisieren soll, den die Bankkaufleute gerade beim Interview vortragen. Schnell 'materialisiert' sich der Gedanke dann im ersten Prototypen, die Betroffenen sind ob der Symbolkraft des technischen Geräts beeindruckt, und so pflanzt sich der Gestaltungsfehler durch alle Systemversionen fort: Das Icon differenzierte nicht und muß durch Text ergänzt werden.

Gestaltung ist orthogonal zu allen Aktivitäten im Softwareentwicklungsprozeß und 'passiert' oder findet bewußt statt zu jedem Zeitpunkt, an dem wir mit Spielräumen, mit Unsicherheit, wie es Nake nennt, umgehen dürfen. Dieses Geschenk des Umgehen-Dürfens mit Spielräumen nimmt uns glücklicherweise keine Norm und kein Styleguide. Unglücklicherweise ist damit auch die Freiheit verbunden, Fehler machen zu dürfen. Und ein zweites: Unsere stärksten 'Waffen' bei der Bewältigung von Komplexität sind Abstraktion und Dekomposition. Im Prozeß der Problembewältigung lösen wir uns von vermeintlich unwichtigen, Benutzern aber bedeutungsvollen Details, wir dekomponieren das problembeladene Ganze in seine bewältigbaren Teile. Das 'Look' und manchmal auch das 'Feel' von Fenstern, Radioknöpfen und Schiebern vermitteln uns Styleguides, die gute Gestalt des Ganzen nicht. Als Informatiker haben wir gelernt die Töne zu setzen, nicht aber, wie daraus die (nicht nur funktionale) Melodie wird. Wir sind der berechenbaren Funktion verpflichtet und verlieren darüber das Gefühl für die Ästhetik. Vielleicht sollten wir doch einmal Christian von Ehrenfelsens Arbeit »Über 'Gestaltqualitäten'« lesen.

Von anderen Gestaltungsdisziplinen können wir lernen, daß jede Gestaltungssituation neu ist, neue Anforderungen an den Gestalter stellt und von neuen, oft wider-

sprüchlichen Anforderungen seitens der Anwender und Benutzer begleitet wird. Rahmenbedingungen und Kontexte, innerhalb derer gestaltet werden soll, ändern sich ebenso wie die Anwendungsbereiche. Die Situation, in der gestaltet werden muß, mithin Gestaltung, ist immer konkret; die Gestaltungsbegriffe und -kriterien, die dem Gestalter Leitlinie sein sollen, hingegen sind abstrakt. Begriffe und Gestaltungsgrundsätze können Richtschnur sein, verhindern aber schlechte Gestalt nicht.

Gestaltungsaufgaben sind komplexe Aufgaben, weil heterogene und zum Teil divergierende Anforderungen im Prozeß zusammengeführt werden müssen, und weil sich Gestaltung in einem Bereich immer auch auf andere Bereiche auswirkt. Man kann (noch) kein Informatiksystem spurlos in eine Arbeits- oder Alltagssituation integrieren, Marc Weisers Vorstellung vom »getting the computer out of the way« ist noch Vision; immer sind mit der Benutzung eines Computers auch Veränderungen an Inhalten, Abläufen, an Qualifikationsanforderungen etc. verbunden. Walter Volpert hat unterstrichen, daß man "nicht nicht gestalten" kann, daß mit der Einführung eines Informatiksystems immer Eingriffe in Arbeits- und Lebensverhältnisse verbunden sind (Volpert 1993, S. 59).

An dieser Stelle spanne ich den Bogen von der Gestaltungsdiskussion zurück zu Lehre und Forschung. Denn den Hinweisen und Warnungen der auf Analyse spezialisierten Sozialwissenschaftler können wir, der Synthese verpflichteten Gestalter stolz das Zwiebelmodell, die Leavitt-Raute und das magische Dreieck von Kubicek und Taube entgegenhalten: Wir haben diese Interdependenzen bedacht und in unseren Schaubildern vorgesehen. In unseren Vorgehensmodellen haben wir alle notwendigen Aktivitäten, deren Ergebnistypen und die Interventionspunkte vorgesehen. Wir könnten zufrieden sein, wenn damit nicht die Lösung des Problems nur vorgetäuscht würde: »What does make design a problem in real world cases is that we are trying to make a diagram for forces whose field we do not understand« (Alexander 1964, p. 21). Um sie zu verstehen, benötigen wir nicht die Gestalter aus den Anwendungsbereichen sondern die Analytiker aus den anderen Disziplinen.

8 Literatur

Alexander, Ch.: Notes on the Synthesis of Form, Cambridge, MA 1964

Capurro, R. et al.: Ethische Leitlinien der Gesellschaft für Informatik, Informatik-Spektrum 16 (1993), S. 238-240

Coy, W.: Brauchen wir eine Theorie der Informatik?, Informatik-Spektrum 12 (1989), S. 256-266

Coy, W.: Computer als Medium - Drei Aufsätze, Universität Bremen, Fachbereich Mathematik/ Informatik, Bericht 3/94, Bremen 1994

Denning, P.J. et al.: Computing as a Discipline, Communications of the ACM 32 (1989), pp. 9—23

Gorny, P.: Zwischenbilanz Menschengerechte Gestaltung von Software, Universität Oldenburg, Fachbereich Informatik, Oldenburg 1992

Gütegemeinschaft Software e.V.: Das Anwenderbrevier - Die zehn Tugenden guter Software, Frankfurt 1987

Hacker, W.: Software-Ergonomie: Gestalten rechnergestützter Arbeit?!, in: Schönpflug, W. und M. Wittstock (Hrsg.), Software-Ergonomie '87, Stuttgart 1987, S. 31-45

Hellige, H.D.: Technikgestaltung: Ein Begriff als Programm. Geschichte, Systematik und Probleme, Universität Bremen, Forschungszentrum Arbeit und Technik, Bremen 1994

IBM: Systems Application Architecture: Common User Access - Guide to User Interface Design, Cary, NC 1991

IBM: Systems Application Architecture: Common User Access - Advanced Interface Design Refernce, Cary, NC 1991

Kubicek, H.: Defizite der Informatik hinsichtlich ihrer Folgenbewältigung und die politische Bewältigung dieser Defizite, Universität Trier, Trier 1984

Kubicek, H. und W. Taube: Die gelegentlichen Nutzer als Herausforderung für die Systementwicklung, Informatik-Spektrum 17 (1994) Nr.6 (im Druck)

Lauter, B.: Software-Ergonomie in der Praxis, München 1987

Nachreiner, F.: Methodenprobleme der Software-Ergonomie, in: Hartmann, A. et al. (Hrsg.), Menschengerechte Groupware - Software-ergonomische Gestaltung und partizipative Umsetzung, Stuttgart 1994, S. 51-63

Nachreiner, F. und E. Mesenholl: Defizite der Software-Ergonomie. Ergebnisse einer Bilanzierung vorliegender Forschungsergebnisse zur Arbeit an Bildschirmgeräten, in: Coy, W. et al. (Hrsg.), Menschengerechte Software als Wettbewerbsfaktor, Stuttgart 1993, S. 602-615

Nake, F.: Informatik und die Maschinisierung von Kopfarbeit, in: Coy, W. et al. (Hrsg.), Sichtweisen der Informatik, Braunschweig 1992, S. 181-201

Norman, D.A.: Dinge des Alltags - Gutes Design und Psychologie für Gebrauchsgegenstände, Frankfurt 1989

Oberquelle, H.: MCI - Quo vadis ? Perspektiven für die Gestaltung und Entwicklung der Mensch-Computer-Interaktion, in: Ackermann, D. und E. Ulich (Hrsg.), Software-Ergonomie '91, Stuttgart 1991, S. 9-24

Oberquelle, H.: Software-Ergonomie lehren - Software-Ergonomie lernen, Ergonomie & Informatik (1994) Nr. 22, S. 24-26

Rödiger, K.-H.: Beiträge der Softwareergonomie zu den frühen Phasen der Softwareentwicklung, in: Bullinger, H.-J. (Hrsg.), Software-Ergonomie '85, Stuttgart 1985, S. 455-464

Rödiger, K.-H.: Einschränkung des Handlungsspielraums durch den Einsatz von Dialogsystemen - Ergebnisse einer Felduntersuchung, in: Nullmeier, E. und K.-H. Rödiger (Hrsg.), Dialogsysteme in der Arbeitswelt, Mannheim 1987, S. 229-244

Rolf, A.: Informatik als Gestaltungswissenschaft - Bausteine für einen Sichtwechsel, in: Langenheder, W., G. Müller und B. Schinzel (Hrsg.), Informatik cui bono?, Berlin 1992, S. 40-48

Rosebrock, J. und M. Siegler: Bewertung des Common User Access (CUA), Diplomarbeit, Universität Bremen, Fachbereich Mathematik/Informatik, Bremen 1993

Siefkes, D.: Sinn im Formalen? Wie wir mit Maschinen und Formalismen umgehen, in: Coy, W. et al. (Hrsg.), Sichtweisen der Informatik, Braunschweig 1992, S. 97-114

Smith, D.C., C. Irby, R. Kimball, B. Verplank, and E. Harslem: Designing the Star User Interface, Byte (1982) No. 4, pp. 242-282

Spillner, A.: Kann eine Krise 25 Jahre dauern?, Informatik-Spektrum 17 (1994), S. 48-52

Szczepanek, U.: Bewertung des Windows Application Design Guide, Diplomarbeit, Universität Bremen, Fachbereich Mathematik/Informatik, Bremen 1994

Thimbleby, H.: User Interface Design, Wokingham 1991

Tognazzini, B.: Tog on Interface, Reading, MA 1992

Volpert, W.: Erhalten und Gestalten - Von der notwendigen Zähmung des Gestaltungsdrangs, in: Coy, W. et al. (Hrsg.), Sichtweisen der Informatik, Braunschweig 1992, S. 171-180

Volpert, W.: Von der Software-Ergonomie zur Arbeitsinformatik, in: Rödiger, K.-H. (Hrsg.), Software-Ergonomie '93, Stuttgart 1993, S. 51-65

Weiser, M.: Some Computer Science Issues in Ubiquitous Computing, Communications of the ACM 36 (1993) No. 7, pp. 75-85

Weltz, F. und R.G. Ortmann: Das Softwareprojekt - Projektmanagement in der Praxis, Frankfurt 1992